自动化测试框架 pytest 教程 1-快速入门

python 自动化测试人工智能

- 测试
- 测试框架
- common
- 机器学习
- linux
- python
- 数据分析

本书还在紧张整理中，完整目录：自动化测试框架 pytest 教程
作者：磁针石 版本号 V0.1 2018-05-17

- 第 1 章 快速入门
- 第 2 章 测试函数
- 第 3 章 pytest Fixtures
- 第 4 章 内置 Fixtures
- 第 5 章 插件
- 第 6 章 配置
- 第 7 章 与其他工具集成
- 第 8 章 更多插件
- 第 9 章 应用实例
- 第 10 章 附录 1：python 打包
- 第 11 章 附录 2：单元测试和 xunit 介绍

目录

什么是 pytest？

pytest 是一个强大的 Python 测试工具，它可以用于所有类型和级别的软件测试。Pytest 可以被开发团队，QA 团队，独立测试小组，实践 TDD 的个人和开放源代码项目。实际上，整个互联网上的项目都是从 unittest 或者 nose 转向 pytest，包括 Mozilla 和 Dropbox。为什么？因为 pytest 提供 强大的功能，如'断言'重写，第三方插件模型，强大但简单的 fixture 模型。

pytest 是软件测试框架，这意味着 pytest 是命令行工具。它会自动找到你写的测试，运行测试并报告结果。可编写插件或安装第三方来扩展插件。它可以用来测试 Python 发行版。它很容易与其他工具对接，如持续集成和网页自动化。

Pytest 脱颖而出的原因：

- 简单
- 易读
- 用 assert 来测试失败，而不是 self.assertEqual() 或者 self.assertLessThan()
- 可运行 unittest 或 nose 测试。

事实上很多自动化测试平台，底层就是用驱动的。它们用 flask 或 django 等提供友好的页面展示，但是核心层还是在 pytest 和一些测试库的开发。

本文基于　ubuntu 16，　python3 书写，但是在其他平台通常也应该可以执行。

技术支持 QQ 群：　144081101　591302926 567351477

接口自动化性能测试数据分析人工智能从业专家一对一线上培训大纲

本文最新版本

```
# pip3 install pytest
Collecting pytest
  Downloading
https://files.pythonhosted.org/packages/76/52/fc48d02492d9e6070cb672d91
33382e83084f567f88eff1c27bd2c6c27a8/pytest-3.5.1-py2.py3-none-any.whl
(192kB)
    100% |████████████████████████████████| 194kB 992kB/s
Requirement already satisfied: setuptools in /usr/lib/python3/dist-
packages (from pytest) (20.7.0)
Collecting pluggy<0.7,>=0.5 (from pytest)
  Downloading
https://files.pythonhosted.org/packages/ba/65/ded3bc40bbf8d887f262f150f
be1ae6637765b5c9534bd55690ed2c0b0f7/pluggy-0.6.0-py3-none-any.whl
Collecting more-itertools>=4.0.0 (from pytest)
  Downloading
https://files.pythonhosted.org/packages/7a/46/886917c6a4ce49dd3fff250c0
```

```
1c5abac5390d57992751384fe61befc4877/more_itertools-4.1.0-py3-none-
any.whl (47kB)
    100% |████████████████████████████████| 51kB 2.3MB/s
Collecting attrs>=17.4.0 (from pytest)
  Downloading
https://files.pythonhosted.org/packages/41/59/cedf87e91ed541be7957c501a
92102f9cc6363c623a7666d69d51c78ac5b/attrs-18.1.0-py2.py3-none-any.whl
Requirement already satisfied: six>=1.10.0 in /usr/lib/python3/dist-
packages (from pytest) (1.10.0)
Collecting py>=1.5.0 (from pytest)
  Downloading
https://files.pythonhosted.org/packages/67/a5/f77982214dd4c8fd104b066f2
49adea2c49e25e8703d284382eb5e9ab35a/py-1.5.3-py2.py3-none-any.whl  (84kB)
    100% |████████████████████████████████| 92kB 2.6MB/s
tensorflow-tensorboard 1.5.1 has requirement bleach==1.5.0, but you'll
have bleach 2.1.3 which is incompatible.
tensorflow-tensorboard 1.5.1 has requirement html5lib==0.9999999, but
you'll have html5lib 1.0.1 which is incompatible.
Installing collected packages: pluggy, more-itertools, attrs, py,
pytest
Successfully installed attrs-18.1.0 more-itertools-4.1.0 pluggy-0.6.0
py-1.5.3 pytest-3.5.1

$ pytest -h # 查看帮助
usage: pytest [options] [file_or_dir] [file_or_dir] [...]

positional arguments:
  file_or_dir

general:
  -k EXPRESSION         only run tests which match the given substring
                        expression. An expression is a python
evaluatable
                        expression where all names are substring-
matched
                        against test names and their parent classes.
Example:
                        -k 'test_method or test_other' matches all test
                        functions and classes whose name contains
                        'test_method' or 'test_other', while -k 'not
                        test_method' matches those that don't contain
                        'test_method' in their names. Additionally
keywords
                        are matched to classes and functions containing
extra
                        names in their 'extra_keyword_matches' set, as
well as
                        functions which have names assigned directly to
them.
  -m MARKEXPR           only run tests matching given mark expression.
                        example: -m 'mark1 and not mark2'.
  --markers             show markers (builtin, plugin and per-project
ones).
  -x, --exitfirst       exit instantly on first error or failed test.
  --maxfail=num         exit after first num failures or errors.
  --strict              marks not registered in configuration file
raise
```

```
                        errors.
  -c file               load configuration from `file` instead of
trying to
                        locate one of the implicit configuration files.
  --continue-on-collection-errors
                        Force test execution even if collection errors
occur.
  --rootdir=ROOTDIR     Define root directory for tests. Can be
relative path:
                        'root_dir', './root_dir',
'root_dir/another_dir/';
                        absolute path: '/home/user/root_dir'; path with
                        variables: '$HOME/root_dir'.
  --fixtures, --funcargs
                        show available fixtures, sorted by plugin
appearance
                        (fixtures with leading '_' are only shown with
'-v')
  --fixtures-per-test   show fixtures per test
  --import-mode={prepend,append}
                        prepend/append to sys.path when importing test
                        modules, default is to prepend.
  --pdb                  start the interactive Python debugger on errors.
  --pdbcls=modulename:classname
                        start a custom interactive Python debugger on
errors.
                        For example:
                        --pdbcls=IPython.terminal.debugger:TerminalPdb
  --capture=method      per-test capturing method: one of fd|sys|no.
  -s                    shortcut for --capture=no.
  --runxfail            run tests even if they are marked xfail
  --lf, --last-failed   rerun only the tests that failed at the last
run (or
                        all if none failed)
  --ff, --failed-first  run all tests but run the last failures first.
This
                        may re-order tests and thus lead to repeated
fixture
                        setup/teardown
  --nf, --new-first     run tests from new files first, then the rest
of the
                        tests sorted by file mtime
  --cache-show          show cache contents, don't perform collection
or tests
  --cache-clear         remove all cache contents at start of test run.
  --lfnf={all,none}, --last-failed-no-failures={all,none}
                        change the behavior when no test failed in the
last
                        run or no information about the last failures
was
                        found in the cache

reporting:
  -v, --verbose         increase verbosity.
  -q, --quiet           decrease verbosity.
  --verbosity=VERBOSE   set verbosity
  -r chars              show extra test summary info as specified by
chars
                        (f)ailed, (E)error, (s)skipped, (x)failed,
(X)passed,
                        (p)passed, (P)passed with output, (a)all except
pP.
                        Warnings are displayed at all times except when
                        --disable-warnings is set
  --disable-warnings, --disable-pytest-warnings
                        disable warnings summary
  -l, --showlocals       show locals in tracebacks (disabled by default).
  --tb=style            traceback print mode
(auto/long/short/line/native/no).
  --show-capture={no,stdout,stderr,log,all}
                        Controls how captured stdout/stderr/log is
shown on
                        failed tests. Default is 'all'.
  --full-trace          don't cut any tracebacks (default is to cut).
  --color=color         color terminal output (yes/no/auto).
  --durations=N         show N slowest setup/test durations (N=0 for
all).
  --pastebin=mode       send failed|all info to bpaste.net pastebin
service.
  --junit-xml=path      create junit-xml style report file at given
path.
  --junit-prefix=str    prepend prefix to classnames in junit-xml
output
  --result-log=path      DEPRECATED path for machine-readable result log.

collection:
  --collect-only        only collect tests, don't execute them.
  --pyargs              try to interpret all arguments as python
packages.
  --ignore=path         ignore path during collection (multi-allowed).
  --deselect=nodeid_prefix
                        deselect item during collection (multi-allowed).
  --confcutdir=dir      only load conftest.py's relative to specified
dir.
  --noconftest          Don't load any conftest.py files.
  --keep-duplicates     Keep duplicate tests.
  --collect-in-virtualenv
                        Don't ignore tests in a local virtualenv
directory
  --doctest-modules     run doctests in all .py modules
  --doctest-report={none,cdiff,ndiff,udiff,only_first_failure}
                        choose another output format for diffs on
doctest
                        failure
  --doctest-glob=pat    doctests file matching pattern, default:
test*.txt
  --doctest-ignore-import-errors
                        ignore doctest ImportErrors
  --doctest-continue-on-failure
                        for a given doctest, continue to run after the
first
                        failure
```

```
test session debugging and configuration:
  --basetemp=dir        base temporary directory for this test run.
  --version             display pytest lib version and import
information.
  -h, --help            show help message and configuration info
  -p name               early-load given plugin (multi-allowed). To
avoid
                        loading of plugins, use the `no:` prefix, e.g.
                        `no:doctest`.
  --trace-config        trace considerations of conftest.py files.
  --debug               store internal tracing debug information in
                        'pytestdebug.log'.
  -o OVERRIDE_INI, --override-ini=OVERRIDE_INI
                        override ini option with "option=value" style,
e.g.
                        `-o xfail_strict=True -o cache_dir=cache`.
  --assert=MODE         Control assertion debugging tools. 'plain'
performs no
                        assertion debugging. 'rewrite' (the default)
rewrites
                        assert statements in test modules on import to
provide
                        assert expression information.
  --setup-only          only setup fixtures, do not execute tests.
  --setup-show          show setup of fixtures while executing tests.
  --setup-plan          show what fixtures and tests would be executed
but
                        don't execute anything.

pytest-warnings:
  -W PYTHONWARNINGS, --pythonwarnings=PYTHONWARNINGS
                        set which warnings to report, see -W option of
python
                        itself.

logging:
  --no-print-logs       disable printing caught logs on failed tests.
  --log-level=LOG_LEVEL
                        logging level used by the logging module
  --log-format=LOG_FORMAT
                        log format as used by the logging module.
  --log-date-format=LOG_DATE_FORMAT
                        log date format as used by the logging module.
  --log-cli-level=LOG_CLI_LEVEL
                        cli logging level.
  --log-cli-format=LOG_CLI_FORMAT
                        log format as used by the logging module.
  --log-cli-date-format=LOG_CLI_DATE_FORMAT
                        log date format as used by the logging module.
  --log-file=LOG_FILE   path to a file when logging will be written to.
  --log-file-level=LOG_FILE_LEVEL
                        log file logging level.
  --log-file-format=LOG_FILE_FORMAT
                        log format as used by the logging module.
  --log-file-date-format=LOG_FILE_DATE_FORMAT
                        log date format as used by the logging module.
```

```
[pytest] ini-options in the first pytest.ini|tox.ini|setup.cfg file
found:

  markers (linelist)        markers for test functions
  empty_parameter_set_mark (string) default marker for empty
parametersets
  norecursedirs (args)      directory patterns to avoid for recursion
  testpaths (args)          directories to search for tests when no
files or directories are given in the command line.
  console_output_style (string) console output: classic or with
additional progress information (classic|progress).
  usefixtures (args)        list of default fixtures to be used with
this project
  python_files (args)       glob-style file patterns for Python test
module discovery
  python_classes (args)     prefixes or glob names for Python test class
discovery
  python_functions (args)   prefixes or glob names for Python test
function and method discovery
  xfail_strict (bool)       default for the strict parameter of xfail
markers when not given explicitly (default: False)
  junit_suite_name (string) Test suite name for JUnit report
  junit_logging (string)    Write captured log messages to JUnit report:
one of no|system-out|system-err
  doctest_optionflags (args) option flags for doctests
  doctest_encoding (string) encoding used for doctest files
  cache_dir (string)        cache directory path.
  filterwarnings (linelist) Each line specifies a pattern for
warnings.filterwarnings. Processed after -W and --pythonwarnings.
  log_print (bool)          default value for --no-print-logs
  log_level (string)        default value for --log-level
  log_format (string)       default value for --log-format
  log_date_format (string) default value for --log-date-format
  log_cli (bool)            enable log display during test run (also
known as "live logging").
  log_cli_level (string)    default value for --log-cli-level
  log_cli_format (string)   default value for --log-cli-format
  log_cli_date_format (string) default value for --log-cli-date-format
  log_file (string)         default value for --log-file
  log_file_level (string)   default value for --log-file-level
  log_file_format (string) default value for --log-file-format
  log_file_date_format (string) default value for --log-file-date-
format
  addopts (args)            extra command line options
  minversion (string)       minimally required pytest version

environment variables:
  PYTEST_ADDOPTS            extra command line options
  PYTEST_PLUGINS            comma-separated plugins to load during
startup
  PYTEST_DEBUG              set to enable debug tracing of pytest's
internals


to see available markers type: pytest --markers
to see available fixtures type: pytest --fixtures
```

```
(shown according to specified file_or_dir or current dir if not
specified; fixtures with leading '_' are only shown with the '-v'
option
```

快速入门

```
def test_passing():
    assert (1, 2, 3) == (1, 2, 3)
```

执行：

```
$ pytest pass_test.py

============================================== test session starts =====
========================================

platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0

rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:

collected 1 item


pass_test.py .                                        [100%]


========================================== 1 passed in 0.01 seconds ===
========================================
```

pass_test.py 后的点表示一个测试运行并通过。 如果你需要更多信息，您可以使用-v 或--verbose

```
$ pytest pass_test.py  -v
=========================================== test session starts
===========================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0 -
- /usr/bin/python3
cachedir: .pytest_cache
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 1 item

pass_test.py ::test_passing PASSED
[100%]

======================================== 1 passed in 0.01 seconds
=========================================
```

在彩色终端，PASSED 和底线是绿色的。

```
def test_failing():
    assert (1, 2, 3) == (3, 2, 1)
```

执行：

```
$ pytest fail_test.py
=========================================== test session starts
===========================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 1 item

fail_test.py F
[100%]

================================================ FAILURES
=================================================
______________________________________________ test_failing
_______________________________________________

    def test_failing():
>       assert (1, 2, 3) == (3, 2, 1)
E       assert (1, 2, 3) == (3, 2, 1)
E         At index 0 diff: 1 != 3
E         Use -v to get the full diff

fail_test.py :2: AssertionError
======================================== 1 failed in 0.03 seconds
=========================================

$ pytest fail_test.py -v
=========================================== test session starts
===========================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0 -
- /usr/bin/python3
cachedir: .pytest_cache
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 1 item

fail_test.py ::test_failing FAILED
[100%]

================================================ FAILURES
=================================================
______________________________________________ test_failing
_______________________________________________
```

```
    def test_failing():
>       assert (1, 2, 3) == (3, 2, 1)
E       assert (1, 2, 3) == (3, 2, 1)
E         At index 0 diff: 1 != 3
E         Full diff:
E         - (1, 2, 3)
E         ?  ^     ^
E         + (3, 2, 1)
E         ?  ^     ^

fail_test.py :2: AssertionError
======================================== 1 failed in 0.02 seconds
=========================================
```

如果没有参数，pytest 会查看当前目录和所有子目录的测试文件(test_开头或者_test 结尾)并运行。 也可以指定文件名，目录名称或这些名称的列表。

我们创建 tasks 的子目录，增加几个测试

task1_test.py

```
from collections import namedtuple

Task = namedtuple('Task', ['summary', 'owner', 'done', 'id'])
Task.__new__.__defaults__ = (None, None, False, None)


def test_defaults():
    """Using no parameters should invoke defaults."""
    t1 = Task()
    t2 = Task(None, None, False, None)
    assert t1 == t2


def test_member_access():
    """Check .field functionality of namedtuple."""
    t = Task('buy milk', 'brian')
    assert t.summary == 'buy milk'
    assert t.owner == 'brian'
    assert (t.done, t.id) == (False, None)
```

task2_test.py

```
from collections import namedtuple


Task = namedtuple('Task', ['summary', 'owner', 'done', 'id'])
Task.__new__.__defaults__ = (None, None, False, None)
```

```
def test_asdict():
    """_asdict() should return a dictionary."""
    t_task = Task('do something', 'okken', True, 21)
    t_dict = t_task._asdict()
    expected = {'summary': 'do something',
                'owner': 'okken',
                'done': True,
                'id': 21}
    assert t_dict == expected


def test_replace():
    """replace() should change passed in fields."""
    t_before = Task('finish book', 'brian', False)
    t_after = t_before._replace(id=10, done=True)
    t_expected = Task('finish book', 'brian', True, 10)
    assert t_after == t_expected
```

执行示例：

```
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$ pytest
============================================ test session starts
============================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 6 items

fail_test.py F
[ 16%]
pass_test.py .
[ 33%]
tasks/task1_test.py ..
[ 66%]
tasks/task2_test.py ..
[100%]

================================================= FAILURES
==================================================
______________________________________________ test_failing
_______________________________________________

    def test_failing():
>       assert (1, 2, 3) == (3, 2, 1)
E       assert (1, 2, 3) == (3, 2, 1)
E         At index 0 diff: 1 != 3
E         Use -v to get the full diff

fail_test.py:2: AssertionError
==================================== 1 failed, 5 passed in 0.04 seconds
====================================
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$ pytest
tasks
```

```
=========================================== test session starts
===========================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 4 items

tasks/task1_test.py ..
[ 50%]
tasks/task2_test.py ..
[100%]

======================================== 4 passed in 0.02 seconds
=========================================
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$ pytest -
v tasks/task2_test.py::test_asdict
=========================================== test session starts
===========================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0 --
/usr/bin/python3
cachedir: .pytest_cache
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 1 item

tasks/task2_test.py::test_asdict PASSED
[100%]

======================================== 1 passed in 0.01 seconds
=========================================
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$ pytest -
-collect-only
=========================================== test session starts
===========================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 6 items
<Module 'fail_test.py'>
  <Function 'test_failing'>
<Module 'pass_test.py'>
  <Function 'test_passing'>
<Module 'tasks/task1_test.py'>
  <Function 'test_defaults'>
  <Function 'test_member_access'>
<Module 'tasks/task2_test.py'>
  <Function 'test_asdict'>
  <Function 'test_replace'>

====================================== no tests ran in 0.02 seconds
=======================================
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$ pytest -
-collect-only -k "asdict or defaults"
=========================================== test session starts
===========================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 6 items / 4 deselected
<Module 'tasks/task1_test.py'>
  <Function 'test_defaults'>
```

```
<Module 'tasks/task2_test.py'>
  <Function 'test_asdict'>

======================================= 4 deselected in 0.02 seconds
========================================
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$ pytest -
k "asdict or defaults"
============================================ test session starts
============================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 6 items / 4 deselected

tasks/task1_test.py .
[ 50%]
tasks/task2_test.py .
[100%]

================================== 2 passed, 4 deselected in 0.02
seconds ==================================
```

发现规则小结

- 测试文件应该命名为 test_.py 或_test.py
- 测试方法和函数应该被命名为 test_。
- 测试类应该被命名为 Test

结果类型:

以下是测试功能的可能结果:

- PASSED (.): 测试成功。
- FAILED (F):测试失败(或 XPASS + strict)。
- SKIPPED (s): 测试被跳过。 你可以使用@pytest.mark.skip()或 pytest.mark.skipif()修饰器告诉 pytest 跳过测试
- xfail (x): 预期测试失败。@pytest.mark.xfail()
- XPASS (X): 测试不应该通过。
- ERROR (E): 错误

更多选项

- marker 标签

比如只执行 test_replace()和 test_member_access()

```
import pytest
```

```
...
@pytest.mark.run_these_please
def test_member_access():
...
```

test_replace()也进行同样的修改

```
$  cd  /path/to/code/ch1/tasks
 pytest  -v  -m  run_these_please
$
================== test session starts ====================
collected 4 items
task2.py::test_replace PASSED
task1.py::test_member_access PASSED
=================== 2 tests deselected ===================
========= 2 passed, 2 deselected in 0.02 seconds =========
```

更多执行方法：　-m "mark1 and mark2"、 -m "mark1 and not mark2"、-m "mark1 or mark2"

- -x, –exitfirst 失败后停止执行

  首次失败后停止执行：py.test -x

  py.test --maxfail=2 两次失败之后停止执行

```
$ pytest -x
================================================= test session
starts =================================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-
0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 6 items

fail_test.py F

======================================================= FAILURES
=======================================================
____________________________________________________ test_failing
_____________________________________________________

    def test_failing():
>       assert (1, 2, 3) == (3, 2, 1)
E       assert (1, 2, 3) == (3, 2, 1)
E         At index 0 diff: 1 != 3
E         Use -v to get the full diff

fail_test.py:2: AssertionError
============================================== 1 failed in 0.04
```

```
seconds ===============================================
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$

1$ pytest --tb=no
================================================= test session
starts =================================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-
0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 6 items

fail_test.py F
[ 16%]
pass_test.py .
[ 33%]
tasks/task1_test.py ..
[ 66%]
tasks/task2_test.py ..
[100%]

========================================= 1 failed, 5 passed in
0.05 seconds ==========================================
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$

$ pytest --maxfail=1 --tb=no
================================================= test session
starts =================================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-
0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 6 items

fail_test.py F

============================================== 1 failed in 0.04
seconds ===============================================
andrew@andrew-PowerEdge-
T630:~/code/backup/pytest_testing/ch1$ pytest --maxfail=2  --tb=no
================================================= test session
starts =================================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-
0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 6 items

fail_test.py F
[ 16%]
pass_test.py .
[ 33%]
tasks/task1_test.py ..
[ 66%]
tasks/task2_test.py ..
[100%]

========================================= 1 failed, 5 passed in
0.04 seconds ==========================================
```

- 设置捕捉

默认捕捉方式为 file descriptor (FD)级捕捉。捕捉所有到操作系统的 1,2 输出。

syslevel 级捕捉只捕捉 python 的 sys.stdout 和 sys.stderr。

```
py.test -s              # disable all capturing 实际为--capture=no  这样
print 就可以在屏幕输出。
py.test --capture=sys # replace sys.stdout/stderr with in-mem files
py.test --capture=fd  # also point filedescriptors 1 and 2 to temp
file
```

* --lf, --last-failed 执行上次失败的测试

多在--tb　之后使用

--ff / --failed-first 则会先执行失败的，然后执行成功的。

```
$ pytest --lf
================================================== test session starts
==================================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 6 items / 5 deselected
run-last-failure: rerun previous 1 failure

fail_test.py F
[100%]

======================================================= FAILURES
========================================================
____________________________________________________ test_failing
_____________________________________________________

    def test_failing():
>       assert (1, 2, 3) == (3, 2, 1)
E       assert (1, 2, 3) == (3, 2, 1)
E         At index 0 diff: 1 != 3
E         Use -v to get the full diff

fail_test.py:2: AssertionError
======================================= 1 failed, 5 deselected in 0.04
seconds ========================================
$ pytest --ff --tb=no
================================================== test session starts
==================================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
```

```
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 6 items
run-last-failure: rerun previous 1 failure first

fail_test.py F
[ 16%]
pass_test.py .
[ 33%]
tasks/task1_test.py ..
[ 66%]
tasks/task2_test.py ..
[100%]

========================================= 1 failed, 5 passed in 0.04
seconds ==========================================
```

- -q 静默模式

-q, --quiet decrease verbosity.

```
$ pytest -q
F.....
[100%]
====================================================== FAILURES
=======================================================
_____________________________________________________ test_failing
______________________________________________________

    def test_failing():
>       assert (1, 2, 3) == (3, 2, 1)
E       assert (1, 2, 3) == (3, 2, 1)
E         At index 0 diff: 1 != 3
E         Full diff:
E         - (1, 2, 3)
E         ?  ^     ^
E         + (3, 2, 1)
E         ?  ^     ^

fail_test.py:2: AssertionError
1 failed, 5 passed in 0.04 seconds
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$ pytest
-qq
F.....
[100%]
====================================================== FAILURES
=======================================================
_____________________________________________________ test_failing
______________________________________________________

    def test_failing():
>       assert (1, 2, 3) == (3, 2, 1)
E       assert (1, 2, 3) == (3, 2, 1)
E         At index 0 diff: 1 != 3
E         Full diff:
E         - (1, 2, 3)
```

```
E           ?  ^      ^
E           + (3, 2, 1)
E           ?  ^      ^

fail_test.py:2: AssertionError
```

- -l 在 traceback 中显示本地变量

--showlocals 在 traceback 中显示本地变量

修改 test_replace()中的 t_expected = Task('finish book' , 'brian' , True, 10)为 t_expected = Task('finish book' , 'brian' , True, 11)

```
$ pytest tasks
================================================== test session starts
==================================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 4 items

tasks/task1_test.py ..
[ 50%]
tasks/task2_test.py .F
[100%]

======================================================= FAILURES
=======================================================
_____________________________________________________ test_replace
______________________________________________________

    def test_replace():
        """replace() should change passed in fields."""
        t_before = Task('finish book', 'brian', False)
        t_after = t_before._replace(id=10, done=True)
        t_expected = Task('finish book', 'brian', True, 11)
>       assert t_after == t_expected
E       AssertionError: assert Task(summary=...e=True, id=10) ==
Task(summary='...e=True, id=11)
E         At index 3 diff: 10 != 11
E         Use -v to get the full diff

tasks/task2_test.py:26: AssertionError
========================================= 1 failed, 3 passed in 0.04
seconds ==========================================
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$ pytest
tasks -l
================================================== test session starts
==================================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 4 items

tasks/task1_test.py ..
```

```
[ 50%]
tasks/task2_test.py .F
[100%]

======================================================= FAILURES
========================================================
______________________________________________________ test_replace
_______________________________________________________

    def test_replace():
        """replace() should change passed in fields."""
        t_before = Task('finish book', 'brian', False)
        t_after = t_before._replace(id=10, done=True)
        t_expected = Task('finish book', 'brian', True, 11)
>       assert t_after == t_expected
E       AssertionError: assert Task(summary=...e=True, id=10) ==
Task(summary='...e=True, id=11)
E         At index 3 diff: 10 != 11
E         Use -v to get the full diff

t_after    = Task(summary='finish book', owner='brian', done=True,
id=10)
t_before   = Task(summary='finish book', owner='brian', done=False,
id=None)
t_expected = Task(summary='finish book', owner='brian', done=True,
id=11)

tasks/task2_test.py:26: AssertionError
========================================= 1 failed, 3 passed in 0.04
seconds ==========================================
```

- –tb=style

py.test -l 在 traceback 中显示本地变量（快捷方式）

py.test --tb=auto 默认格式，首尾为 long，其他为 short

py.test --tb=long 详细的 traceback 信息格式化形式

py.test --tb=native 标准库格式化形式, 没有额外信息

py.test --tb=short 更短的 traceback 格式

py.test --tb=line 每个错误一行

py.test --tb=no 无 traceback

py.test --full-trace 最详细的格式

实例：

```
$ pytest tasks --tb no
==================================================== test session starts
====================================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 4 items

tasks/task1_test.py ..
[ 50%]
tasks/task2_test.py .F
[100%]

========================================= 1 failed, 3 passed in 0.04
seconds ==========================================
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$ pytest
tasks --tb line
==================================================== test session starts
====================================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 4 items

tasks/task1_test.py ..
[ 50%]
tasks/task2_test.py .F
[100%]

====================================================== FAILURES
=======================================================
/home/andrew/code/backup/pytest_testing/ch1/tasks/task2_test.py:26:
AssertionError: assert Task(summary=...e=True, id=10) ==
Task(summary='...e=True, id=11)
========================================= 1 failed, 3 passed in 0.03
seconds ==========================================
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$ pytest
tasks --tb short
==================================================== test session starts
====================================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 4 items

tasks/task1_test.py ..
[ 50%]
tasks/task2_test.py .F
[100%]

====================================================== FAILURES
=======================================================
____________________________________________________ test_replace
_____________________________________________________
tasks/task2_test.py:26: in test_replace
    assert t_after == t_expected
E   AssertionError: assert Task(summary=...e=True, id=10) ==
Task(summary='...e=True, id=11)
```

```
E      At index 3 diff: 10 != 11
E      Use -v to get the full diff
========================================= 1 failed, 3 passed in 0.04
seconds ==========================================
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$ vi
tasks/task2_test.py
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$ pytest
tasks  --duration=3
================================================= test session starts
=================================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 4 items

tasks/task1_test.py ..
[ 50%]
tasks/task2_test.py ..
[100%]

============================================== slowest 3 test
durations ===============================================
0.00s setup    tasks/task1_test.py::test_defaults
0.00s setup    tasks/task2_test.py::test_asdict
0.00s setup    tasks/task2_test.py::test_replace
============================================== 4 passed in 0.02
seconds ===============================================
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$
```

* –durations=N

统计最慢执行时间

```
andrew@andrew-PowerEdge-T630:~/code/backup/pytest_testing/ch1$ pytest
tasks  --duration=3
================================================= test session starts
=================================================
platform linux -- Python 3.5.2, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
rootdir: /home/andrew/code/backup/pytest_testing/ch1, inifile:
collected 4 items

tasks/task1_test.py ..
[ 50%]
tasks/task2_test.py ..
[100%]

============================================== slowest 3 test
durations ===============================================
0.00s setup    tasks/task1_test.py::test_defaults
0.00s setup    tasks/task2_test.py::test_asdict
0.00s setup    tasks/task2_test.py::test_replace
============================================== 4 passed in 0.02
seconds ===============================================
```