Rocky Enterprise Linux 9.2 Manual Pages on command 'trial3.1'

$ man trial3.1

TRIAL(1)   TRIAL(1)

NAME

    trial - run unit tests

SYNOPSIS

    trial [options] [file | package | module | TestCase | testmethod] ...

    trial --help | -h

DESCRIPTION

    trial loads and executes a suite of unit tests, obtained from modules, packages and files

    listed on the command line.

    trial will take either filenames or fully qualified Python names as arguments.   Thus

    `trial   myproject/foo.py',   `trial myproject.foo' and `trial myproject.foo.SomeTest?

    Case.test_method' are all valid ways to invoke trial. Multiple such arguments are also ac?

    cepted, and their order will determine the order in which the corresponding tests are run.

    After running the given test suite, the default test reporter prints a summary of the test

    run. This consists of the word "PASSED" (if all tests ran as expected) or "FAILED" (if any

    test behaved unexpectedly) followed by a count of the different kinds of test results en?

    countered. The possible kinds of test results includes:

    successes

        Tests that passed all their assertions and completed without error.   These are

        marked "PASSED" in the normal test output.

    failures

        Tests that failed an assertion, called self.fail() or explicitly raised self.fail?

        ureException for some reason. These are marked "FAILED" in the normal test output.

errors Tests that raised an unexpected exception (including AssertionError), tests that

caused the tearDown() method to raise an exception, tests that run for longer than

the timeout interval, tests that caused something to call twisted.python.log.err()

without subsequently calling self.flushLoggedErrors(), tests that leave the reactor

in an unclean state, etc. These are marked "ERROR" in the normal test output.

Note that because errors can be caused after the actual test method returns, it is

possible for a single test to be reported as both an error and a failure, and hence

the total number of test results can be greater than the total number of tests exe?

cuted.

skips Tests that were skipped, usually because of missing dependencies. These are marked

"SKIPPED" in the normal test output.

expectedFailures

Tests that failed, but were expected to fail, usually because the test is for a

feature that hasn't been implemented yet. These are marked "TODO" in the normal

test output.

unexpectedSuccesses

Tests that should have been listed under expectedFailures, except that for some

reason the test succeeded. These are marked "SUCCESS!?!" in the normal test output.

OPTIONS

-j, --jobs number

Set the number of process workers to run. It conflicts with the debug, exitfirst

and profile options.

-b, --debug

Run the tests in a debugger. If that debugger is 'pdb' (which is the default if un?

specified), a `.pdbrc' will be loaded from the current directory if it exists. Also

does post-mortem debugging on exceptions.

-B, --debug-stacktraces

Report Deferred creation and callback stack traces.

--coverage

Generate coverage information in the `coverage' subdirectory of the trial temp di?

rectory (`_trial_temp' by default). For each Python module touched by the execution

of the given tests, a file will be created in the coverage directory named for the

module's fully-qualified name with the suffix `.cover'. For example, because the

trial test runner is written in Python, the coverage directory will almost always contain a file named `twisted.trial.runner.cover'.

Each `.cover' file contains a copy of the Python source of the module in question, with a prefix at the beginning of each line containing coverage information. For lines that are not executable (blank lines, comments, etc.) the prefix is blank. For executable lines that were run in the course of the test suite, the prefix is a number indicating the number of times that line was executed. The string `>>>>>>' prefixes executable lines that were not executed in the course of the test suite. Note that this functionality uses Python's sys.settrace() function, so tests that call sys.settrace() themselves are likely to break trial's coverage functionality.

--debugger

Specifies the debugger to use when the --debug option is passed. The argument should be the fully qualified name of an object that implements the same interface as the standard library's `pdb'.

--disablegc

Disable the garbage collector for the duration of the test run. As each test is run, trial saves the TestResult objects, which means that Python's garbage collec? tor has more non-garbage objects to wade through, making each garbage-collection run slightly slower. Disabling garbage collection entirely will make some test suites complete faster (contrast --force-gc, below), at the cost of increasing (possibly greatly) memory consumption. This option also makes tests slightly more deterministic, which might help debugging in extreme circumstances.

-e, --rterrors

Print tracebacks to standard output as soon as they occur.

--force-gc

Run gc.collect() before and after each test case. This can be used to isolate er? rors that occur when objects get collected. This option would be the default, ex? cept it makes tests run about ten times slower.

-h, --help

Print a usage message to standard output, then exit.

--help-order

Print a list of possible orders that TestCase test methods can be run in, then exit. The orders can be used with the --order option described below.

--help-reporters

    Print a list of valid reporters to standard output, then exit. Reporters can be se?

    lected with the --reporter option described below.

--help-reactors

    Print a list of possible reactors to standard output, then exit. Not all listed re?

    actors are available on every platform. Reactors can be selected with the --reactor

    option described below.

-l, --logfile logfile

    Direct the log to a different file. The default file is `test.log'. logfile is

    relative to _trial_temp.

-n, --dry-run

    Go through all the tests and make them pass without running.

-N, --no-recurse

    By default, trial recurses through packages to find every module inside every sub?

    package. Unless, that is, you specify this option.

--nopm Don't automatically jump into debugger for post-mortem analysis of exceptions.

    Only usable in conjunction with --debug.

--order order

    Specify what order to run the individual test methods within the given TestCases.

    By default, they are run alphabetically. See --help-order for a list of other valid

    values.

--profile

    Run tests under the Python profiler.

-r, --reactor reactor

    Choose which reactor to use. See --help-reactors for a list.

--recursionlimit limit

    Set Python's recursion limit. See sys.setrecursionlimit().

--reporter reporter

    Select the reporter to use for trial's output. Use the --help-reporters option to

    see a list of valid reporters.

-x, --exitfirst

    Stop the test run after the first test which does not succeed. This includes fail?

    ures, errors, or unexpected successes. Won't work with the --jobs option currently.

--spew Print an insanely verbose log of everything that happens. Useful when debugging

    freezes or locks in complex code.

--tbformat format

    Format to display tracebacks with. Acceptable values are `default', `brief' and

    `verbose'. `brief' produces tracebacks that play nicely with Emacs' GUD.

--temp-directory directory

    WARNING: Do not use this option unless you know what you are doing.   By default,

    trial creates a directory called _trial_temp under the current working directory.

    When trial runs, it first deletes this directory, then creates it, then changes

    into the directory to run the tests. The log file and any coverage files are stored

    here. Use this option if you wish to have trial run in a directory other than

    _trial_temp. Be warned, trial will delete the directory before re-creating it.

--testmodule filename

    Ask trial to look into filename and run any tests specified using the Emacs-style

    buffer variable `test-case-name'.

--unclean-warnings

    As of Twisted 8.0, trial will report an error if the reactor is left unclean at the

    end of the test. This option is provided to assist in migrating from Twisted 2.5 to

    Twisted 8.0 and later. Enabling this option will turn the errors into warnings.

-u, --until-failure

    Keep looping the tests until one of them raises an error or a failure.   This is

    particularly useful for reproducing intermittent failures.

--version

    Prints the Twisted version number and exit.

--without-module modulenames

    Simulate the lack of the specified comma-separated list of modules. This makes it

    look like the modules are not present in the system, causing tests to check the be?

    havior for that configuration.

-z, --random seed

    Run the tests in random order using the specified seed. Don't pass this option if

    you also are passing --order.

SEE ALSO

The latest version of the trial documentation can be found at http://twistedma?

trix.com/documents/current/core/howto/testing.html

AUTHOR

Written by Jonathan M. Lange

REPORTING BUGS

To report a bug, visit http://twistedmatrix.com/trac/newticket

COPYRIGHT

Copyright ? 2003-2013 Twisted Matrix Laboratories

This is free software; see the source for copying conditions. There is NO warranty; not

even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.