

NAME

App::Prove::State - State storage for the prove command.

VERSION

Version 3.30

DESCRIPTION

The `prove` command supports a `--state` option that instructs it to store persistent state across runs. This module implements that state and the operations that may be performed on it.

SYNOPSIS

```
# Re-run failed tests
$ prove --state=failed,save -rbv
```

METHODS**Class Methods***new*

Accepts a hashref with the following key/value pairs:

- **store**
The filename of the data store holding the data that `App::Prove::State` reads.
- **extensions** (optional)
The test name extensions. Defaults to `.t`.
- **result_class** (optional)
The name of the `result_class`. Defaults to `App::Prove::State::Result`

result_class

Getter/setter for the name of the class used for tracking test results. This class should either subclass from `App::Prove::State::Result` or provide an identical interface.

extensions

Get or set the list of extensions that files must have in order to be considered tests. Defaults to `['.t']`.

results

Get the results of the last test run. Returns a `result_class()` instance.

commit

Save the test results. Should be called after all tests have run.

Instance Methods*apply_switch*

```
$self->apply_switch('failed,save');
```

Apply a list of switch options to the state, updating the internal object state as a result. Nothing is returned.

Diagnostics: - "Illegal state option: %s"

last

Run in the same order as last time

failed

Run only the failed tests from last time

passed

Run only the passed tests from last time

all

Run all tests in normal order

hot
Run the tests that most recently failed first

todo
Run the tests ordered by number of todos.

slow
Run the tests in slowest to fastest order.

fast
Run test tests in fastest to slowest order.

new
Run the tests in newest to oldest order.

old
Run the tests in oldest to newest order.

save
Save the state on exit.

get_tests
Given a list of args get the names of tests that should run

observe_test
Store the results of a test.

save
Write the state to a file.

load
Load the state from a file