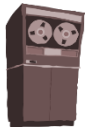

Tape Cheat Sheet

Install

```
npm install tape -g
npm install tape --save-dev
```

Include

```
var test = require( 'tape' ) ;
```

Package.json

```
"devDependencies": {
  "tape": "*"
}
```

Usage

```
test( 'feature', function( assert ) {
  assert.equal( 1, 2 ) ;
  assert.end() ;
}
```

API Reference

assert.plan(n)
Declare that *n* assertions should be run. Call *t.end()* automatically after the *n*th assertion. Any subsequent assertions will generate errors.

assert.end([err])
Declare the end of a test explicitly. Assert that *err* is falsy if supplied.

assert.pass(msg)
Generate a passing assertion with message *msg*.

assert.fail(msg)
Generate a failing assertion with message *msg*.

assert.true(value, [msg])
Assert that *value* is truthy.

assert.false(value, [msg])
Assert that *value* is falsy.

assert.error(err, [msg])
Assert that *err* is falsy. If not, use its err.message as the message.

assert.equal(actual, expected, [msg])
Assert that *actual* === *expected*.

assert.notEqual(actual, expected, [msg])
Assert that *actual* !== *expected*.

assert.deepEqual(actual, expected, [msg])
Assert that *actual* and *expected* have the same structure and nested values with strict comparisons (===) on leaf nodes.

assert.notDeepEqual(actual, expected, [msg])
Assert that *actual* and *expected* do not have the same structure and nested values with strict comparisons (===) on leaf nodes.

assert.deepLooseEqual(actual, expected, [msg])
Assert that *actual* and *expected* have the same structure and nested values with loose comparisons (==) on leaf nodes.

assert.notLooseDeepEqual(actual, expected, [msg])
Assert that *actual* and *expected* do not have the same structure and nested values with loose comparisons (==) on leaf nodes.

API Reference

assert.timeoutAfter(ms)
Automatically timeout the test after *x* milliseconds.

assert.skip(msg)
Generation an assertion that will be skipped over.

assert.throws(fn, expected, [msg])
assert.doesNotThrow(fn, expected, [msg])
Assert that the function call *fn()* throws / does not throw an exception. If specified *expected* must be a RegExp or Function.

assert.test(name, function(assert2))
Create a subtest with new test handle *assert2* inside the current test *assert*. *assert2* will only file when *assert* finishes.

assert.comment(msg)
Print message in sequence with other test output.

test.only(name, function(assert))
Like the standard *test(name, function(assert))* except this is the only test case that will run for the entire process, all other test cases will be ignored.