

NODEJS UNIT TEST

— by [liuxin.rkl](#) 2014-09-09

WHAT UNIT TEST ?

单元测试（又称为模块测试, Unit Testing）是针对程序模块（软件设计的最小单位）来进行正确性检验的测试工作。OOP中的最小单位即方法。

WHY UNIT TEST ?

- 代码质量
- 快速的需求变更
- 重构

HOW UNIT TEST?

```
function division (a, b) {  
  if (b === 0) {  
    return 0;  
  } else {  
    return a/b;  
  }  
}  
  
function test(name, actual, expect) {  
  if (actual === expect) {  
    console.log(name + ' case passed!');  
  } else {  
    console.log(name + ' case failed!');  
  }  
}  
  
test('b != 0', division(10, 5), 2); // b !=0 case passed!  
test('b === 0', division(10, 0), 0);
```

测试框架

Testing / Spec Frameworks

- Mocha BDD or TDD
- Jasmine BDD
- vows BDD
- espresso TDD
- nodeunit TDD
- ...

BDD: Behavior-driven development

```
// The "BDD" interface provides describe(), it(), before(), after(), beforeEac
describe('Array', function(){
  before(function(){
    // ...
  });

  describe('#indexOf()', function(){
    it('should return -1 when not present', function(){
      [1,2,3].indexOf(4).should.equal(-1);
    });
  });
});
```

TDD: Test-driven development

```
// The "TDD" interface provides suite(), test(), setup(), and teardown().
suite('Array', function(){
  setup(function(){
    // ...
  });

  suite('#indexOf()', function(){
    test('should return -1 when not present', function(){
      assert.equal(-1, [1,2,3].indexOf(4));
    });
  });
});
```

BDD和TDD的区别？

区别于TDD，BDD是从user story的角度来编码，在测试中的描述大多是“When [scenario], giving [conditions], it should [expected results]”。

如何选择测试框架？

- 示例完整
- 持续改进
- 容易上手

MOCHA

特性:

- browser support
- use any assertion library you want/任何你想使用的断言库
- extensible reporting, bundled with 9+ reporters/支持9中报告格式，且可扩展
- before, after, before each, after each hooks/钩子
- ...

JASMINE

- 内建的断言匹配方法expect
- custom matchers/自定义的断言匹配
- spies/强大、灵活的spy
- ...

SPY AND MOCK

保证单元测试的独立性，总要用的spy和mock

- [jm-spy](#) jasmine spies module
- [mm](#) mock mate
- [sinonjs](#)

断言库

should BDD style

```
var should = require('should');  
  
(5).should.be.exactly(5).and.be.a.Number;
```

expect BDD style with expect()

```
var expect = require('expect.js');  
  
expect(5).to.be.a('number');
```

chai expect(), assert() and should style

```
var chai = require('chai')
  , expect = chai.expect
  , should = chai.should();
var foo = 'bar';

expect(foo).to.be.a('string');
foo.should.be.a('string');
```


测试覆盖率

- `jscoverage` 行覆盖率
- `istanbul` 行覆盖率、分支覆盖率

实战

code with [mocha、expect、istanbul]

需求：一个web应用-首页展示nodejs测试框架的列表

功能：

- 应用首页
- 获取nodejs测试框架列表的api服务

项目目录

```
// unit-test-demo
.  
├── lib  
│   └── app.js  
├── test  
│   └── app-test.js  
├── Makefile  
├── README.md  
├── index.html  
├── index.js  
└── package.json
```

MAKEFILE

```
BIN_MOCHA = ./node_modules/.bin/mocha
BIN_ISTANBUL = ./node_modules/.bin/istanbul

install:
    @npm install

test:
    @$(BIN_MOCHA) \
        --reporter $(REPORTER) \
        --timeout $(TIMEOUT) \
        $(TESTS)

test-cov:
    @$(BIN_ISTANBUL) cover ./node_modules/.bin/_mocha \
        -- -u exports -R $(REPORTER) -t $(TIMEOUT) \
        $(TESTS)
```

运行测试

```
make test
```

运行测试覆盖率

```
make test-cov
```

行为驱动开发

先写测试: /test/app-test.js

```
var e = require('expect.js');
var request = require('supertest')
var app = require('../lib/app');

describe('Testing / Spec Frameworks app', function () {

  before(function (done) {
    app.listen(1234, done);
  });
  ...
});
```

使用supertest进行http测试

应用首页HTML

```
it('GET / show the title, a loading container', function (done) {  
  request(app)  
    .get('/')  
    .expect(200)  
    .expect('Content-Type', 'text/html')  
    .end(function (err, res) {  
      e(err).to.be(null);  
      var body = res.text;  
      e(body).to.contain('nodejs - Testing / Spec Frameworks');  
      e(body).to.contain('Testing / Spec Frameworks');  
      e(body).to.contain('loading...');  
      done();  
    });  
});
```


获取列表信息的API

```
it('GET /list show the frameworks list, in json data', function (done) {  
  request(app)  
    .get('/list')  
    .expect(200)  
    .expect('Content-Type', 'application/json')  
    .end(function (err, res) {  
      e(err).to.be(null);  
      var body = JSON.parse(res.text);  
      e(body).to.have.length(5);  
      done();  
    });  
});
```

404

```
it('GET /nopage page not found', function (done) {  
  request(app)  
    .get('/nopage')  
    .expect(200)  
    .expect('Content-Type', 'text/html')  
    .end(function (err, res) {  
      e(err).to.be(null);  
      var body = res.text;  
      e(body).to.be('Page not found');  
      done();  
    });  
});
```

LET'S GO ->

Testing / Spec Frameworks app

- 1) "before all" hook
- 2) "after all" hook

0 passing (6ms)

2 failing

1) Testing / Spec Frameworks app "before all" hook:

TypeError: Object #<Object> has no method 'listen'

at Context.<anonymous> (/Users/christine/github/unit-test-demo/test/app-test.js:8:9)

at Hook.Runnable.run (/Users/christine/github/unit-test-demo/node_modules/mocha/lib/runnable.js:

at next (/Users/christine/github/unit-test-demo/node_modules/mocha/lib/runner.js:258:10)

at Object._onImmediate (/Users/christine/github/unit-test-demo/node_modules/mocha/lib/runner.js:

at processImmediate [as _immediateCallback] (timers.js:330:15)

实现APP.JS

```
var app = http.createServer(function (req, res) {
  res.setHeader('Content-Type', 'text/html');
  var urlObj = parse(req.url, true);

  if (urlObj.pathname === '/') {
    var page = fs.readFileSync(path.join(__dirname, '../index.html'), 'utf8');
    res.end(page);
  } else if (urlObj.pathname === '/list') {
    res.setHeader('Content-Type', 'application/json');
    var list = [
      {title: 'mocha', desc: 'simple, flexible, fun javascript test framework for'},
      {title: 'expresso', desc: 'TDD framework by the author of JSpec'},
      {title: 'nodeunit', desc: 'Simple syntax, powerful tools. Based on the ass'},
      {title: 'Vows', desc: 'asynchronous behaviour-driven development for node.'},
      {title: 'others', desc: '...'}
    ];
    res.end(JSON.stringify(list));
  } else {
    res.end('Page not found');
  }
});
```

LET'S GO AGAIN!

Testing / Spec Frameworks app

- ✓ GET / show the title, a loading container
- ✓ GET /nopage page not found
- ✓ GET /list show the frameworks list, in json data

3 passing (35ms)



来跑个覆盖率吧！

```
christine@~/github/unit-test-demo (master) $ make test-cov
```

Testing / Spec Frameworks app

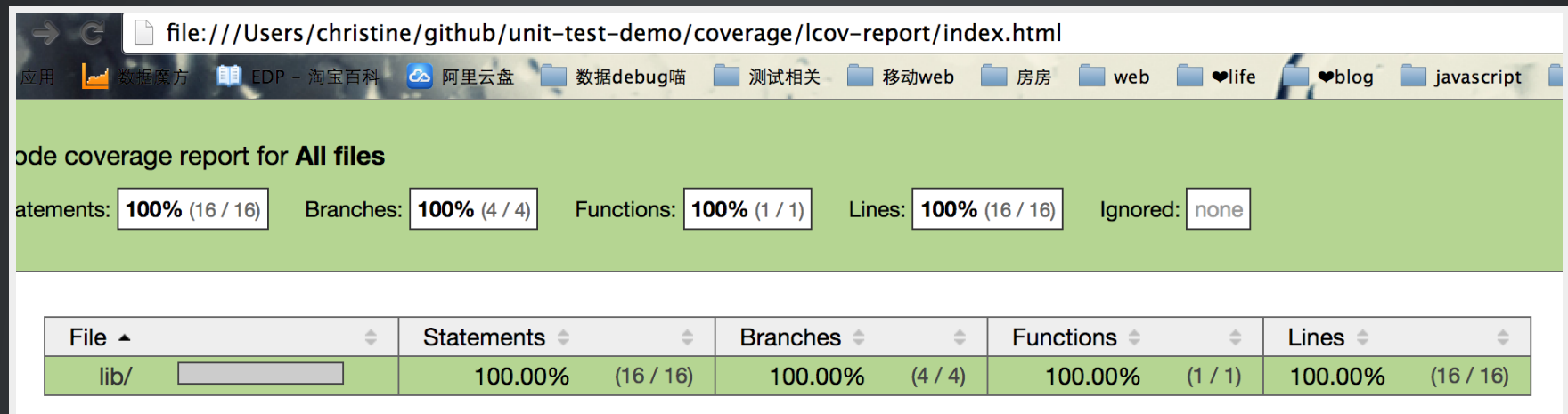
- ✓ GET / show the title, a loading container
- ✓ GET /nopage page not found
- ✓ GET /list show the frameworks list, in json data

3 passing (34ms)

```
=====
Writing coverage object [/Users/christine/github/unit-test-demo/coverage/coverage.json]
Writing coverage reports at [/Users/christine/github/unit-test-demo/coverage]
=====
```

```
===== Coverage summary =====
Statements   : 100% ( 16/16 )
Branches     : 100% ( 4/4 )
Functions    : 100% ( 1/1 )
Lines        : 100% ( 16/16 )
=====
```


详细的HTML COVERAGE REPORT



不持续集成不舒服斯基?

- toast 内部持续集成服务 -> toastman
- travis-ci 开源的持续集成服务

持续集成到TRAVIS-CI

新增配置: .travis.yml

```
language: node_js
node_js:
  - "0.11"
  - "0.10"
  - "0.8"
```

christineRR/unit-test-demo

build passing



demo or examples about nodejs unit test

- Current
- Build History
- Pull Requests
- Branch Summary



master - add build status




#2 passed

ran for 48 sec
about 2 hours ago

 柳心 authored and committed

[Commit 1aecc0d](#)  [Compare c811ae0...1aecc0d](#) 

Build Matrix

Job	Duration	Finished	Node.js
 2.1	16 sec	about 2 hours ago	0.11
 2.2	14 sec	about 2 hours ago	0.10
 2.3	18 sec	about 2 hours ago	0.8

添加BUILD STATUS到项目中

```
[![Build Status](https://travis-ci.org/christineRR/unit-test-demo.svg?branch=m
```

📖 README.md

unit-test-demo build passing

demo or examples about nodejs unit test

完整代码地址戳[这里](#)

参考

- 实战nodejs 单元测试
- the difference between bdd and tdd
- tdd vs bdd
- node modules
- istanbul with mocha

谢谢大家

