JAVASCRIPT

Тестирование кода. Mocha. Chai. Sinon

Unit Testing

Mocha –

многофункциональная платформа(framework) для тестирования JavaScript.

Используется для выполнения тестов

Chai – это библиотека утверждений

Автономные тестовые шпионы (spies), заглушки (stubs) и моки (mocks) для JavaScript.

Используется, чтобы замокать либо застабить(блокировать) зависимости.

Unit Testing

```
$ npm install --global mocha
```

```
$ npm install --save-dev mocha
```

```
$ npm install chai
```

```
npm install sinon
```

Unit Testing

test/testrunner.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Mocha Tests</title>
    <link rel="stylesheet" href="../node_modules/mocha/mocha.css">
</head>
<body>
<div id="mocha"></div>
<script src="https://cdn.rawgit.com/jquery/jquery/2.1.4/dist/jquery.min.js"></script>
<script src="../node_modules/mocha/mocha.js"></script>
<script src="../node_modules/sinon/pkg/sinon.js"></script>
<script src="../node_modules/chai/chai.js"></script>

<script>mocha.setup('bdd')</script>
<script src="specs/test-radio.js"></script>
<script src="specs/test-router.js"></script>

<script>
    mocha.checkLeaks();
    mocha.globals(['jQuery']);
    mocha.run();
</script>

</body>
</html>
```

Unit Testing

Test cases

```
describe('Array', function() {
  // Further code for tests goes here
});
```

```
describe('Array', function() {
  it('should start empty', function() {
    // Test implementation goes here
  });

  // We can have more its here
});
```

```
var assert = chai.assert;
var expect = chai.expect;
var should = chai.should();


describe('Radio', function() {
    describe('propery: topics', function() {
        it('should be an array', function() {
            expect(radio.topics).to.be.an.instanceof(Array);
        });
    });
});
```

Unit Testing

Test cases

```
describe('Radio', function() {
    describe('propery: topics', function() {
        it('should be an array', function() {
            expect(radio.topics).to.be.an.instanceof(Array);
        });
    });
});
```

passes: 1 failures: 0 duration: 0.06s 100%

Radio

propery: topics

✓ should be an array

```
describe('Radio', function() {
    describe('propery: topics', function() {
        it('should be an array', function() {
            expect(radio.topics).to.be.an.instanceof(Function);
        });
    });
});
```

passes: 0 failures: 1 duration: 0.04s 100%

Radio

propery: topics

✗ should be an array

```
AssertionError: expected [] to be an instance of Function
    at Context.<anonymous> (specs/test-radio.js:9:53)
```

Unit Testing

Test cases

```
describe('hooks', function() {

  before(function() {
    // runs before all tests in this block
  });

  after(function() {
    // runs after all tests in this block
  });

  beforeEach(function() {
    // runs before each test in this block
  });

  afterEach(function() {
    // runs after each test in this block
  });

  // test cases
});
```

Unit Testing

Chai

Should

```
chai.should();

foo.should.be.a('string');
foo.should.equal('bar');
foo.should.have.lengthOf(3);
tea.should.have.property('flavors')
  .with.lengthOf(3);
```

Visit Should Guide

Expect

```
var expect = chai.expect;

expect(foo).to.be.a('string');
expect(foo).to.equal('bar');
expect(foo).to.have.lengthOf(3);
expect(tea).to.have.property('flavors')
  .with.lengthOf(3);
```

Visit Expect Guide

Assert

```
var assert = chai.assert;

assert.typeOf(foo, 'string');
assert.equal(foo, 'bar');
assert.lengthOf(foo, 3)
assert.property(tea, 'flavors');
assert.lengthOf(tea.flavors, 3);
```

Visit Assert Guide

Unit Testing

```
var assert = chai.assert;
var expect = chai.expect;
var should = chai.should();


describe('Radio', function () {
    describe('property: topics', function () {
        it('should be an array', function () {
            expect(radio.topics).to.be.an.instanceof(Array);
        });
        it('should be empty', function () {
            expect(radio.topics).to.be.empty;
        });
    });
    describe('method: on', function () {
        it('should create new topic if it does not exist', function () {

        });
    });
});
```

passes: 3 failures: 0 duration: 0.04s 100%

Radio

property: topics

- ✓ should be an array
- ✓ should be empty

method: on

- ✓ should create new topic if it does not exist

Unit Testing

Spies, Stubs and Mocks

Spies

Наиболее распространенные сценарии со шпионами:

- *Проверка того, сколько раз функция вызывалась*
- *Проверка того, какие аргументы были переданы функции*

```
it('should call save once', function() {
  var save = sinon.spy(Database, 'save');

  setupNewUser({ name: 'test' }, function() { });

  save.restore();
  sinon.assert.calledOnce(save);
});
```

Unit Testing

Spies, Stubs and Mocks

Stubs

Наиболее распространенные сценарии с заглушками:

- *для замены проблемных фрагментов кода*
- *чтобы легче тестировать асинхронный код*

```
it('should pass object with correct values to save', function() {
  var save = sinon.stub(Database, 'save');
  var info = { name: 'test' };
  var expectedUser = {
    name: info.name,
    nameLowercase: info.name.toLowerCase()
  };

  setupNewUser(info, function() { });

  save.restore();
  sinon.assert.calledWith(save, expectedUser);
});
```

Unit Testing

Spies, Stubs and Mocks

Mocks

Mocks следует использовать, прежде всего, когда вы будете использовать заглушку, но вам нужно проверить на ней несколько более конкретных действий.

```
it('should pass object with correct values to save only once', function() {
  var info = { name: 'test' };
  var expectedUser = {
    name: info.name,
    nameLowercase: info.name.toLowerCase()
  };
  var database = sinon.mock(Database);
  database.expects('save').once().withArgs(expectedUser);

  setupNewUser(info, function() { });

  database.verify();
  database.restore();
});
```

JAVASCRIPT

Тестирование кода. Mocha. Chai. Sinon