



Testing NightWatch JS

An intro to Automation tests. by David Torroija

Nightwatch logo



Que carajo veremos hoy? Index.html

- What?
- Why Automation?
- Why nightwatch?
- Examples
- CI
- Etc etc.

— Que es esto? (What is this?)

E2e testing, o testing funcional o automated Testing o llamáله como quieras, es un testing que hacés graficamente un robot imitando a un usuario

E2e testing or functional testing or whatever you like to call this, is a testing that simulates a person testing the app

— ¿¿Porqué hacer test funcional??

Why functional test?

— ¿¿Porqué hacer test funcional??

Why functional test?

Siempre caerás a un proyecto que no tiene tests.

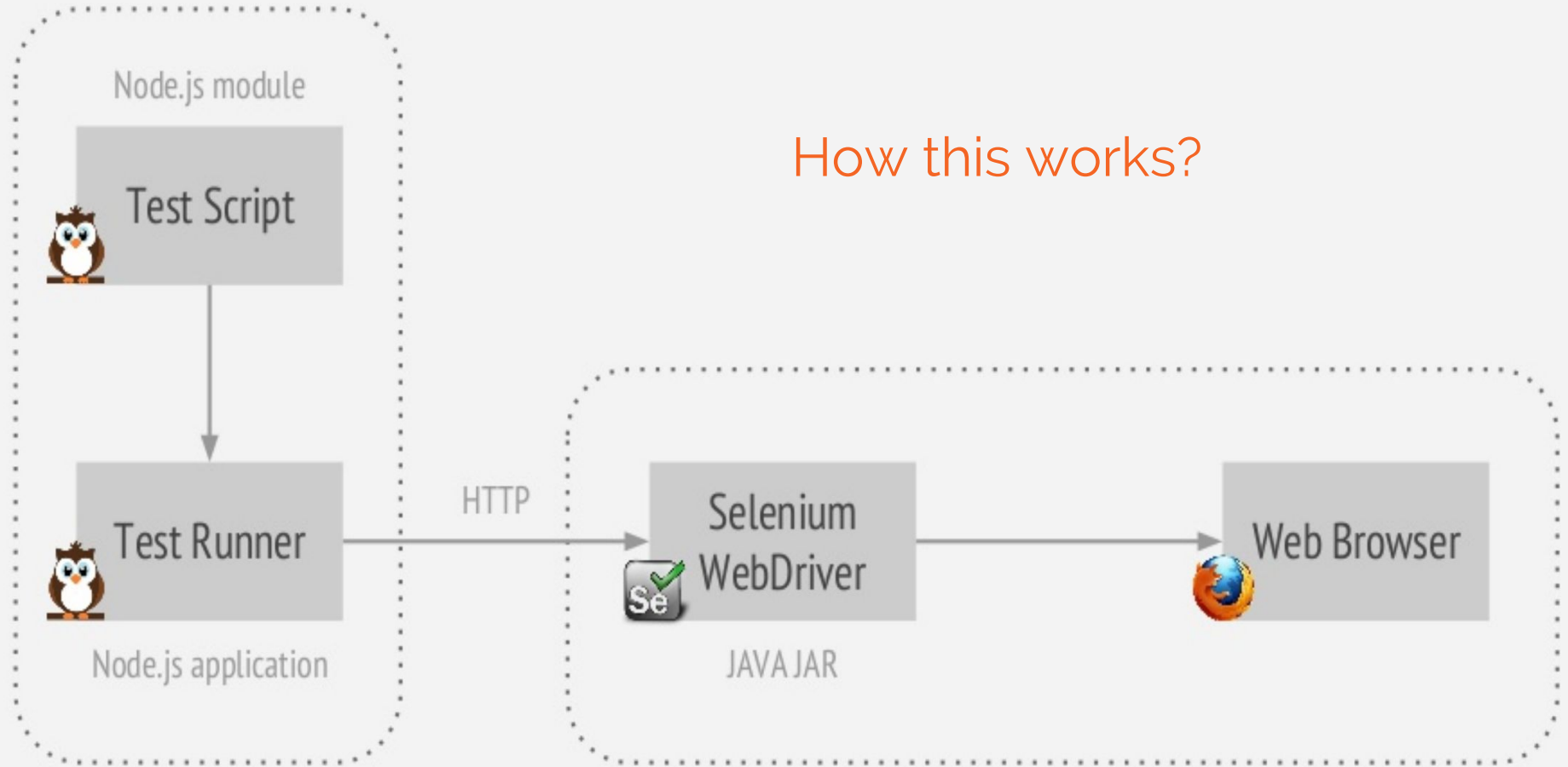
Entonces esta será tu herramienta de batalla para asegurarte que está andando todo y te hará ser el héroe en ese caos.

You will always go to a project that is not properly tested, so this tool is going to help you to be the hero in that caos

¿Why NightwachJS?

- JS == simple
- Extensible
- Uses selenium in the background
- Uses nodejs
- Easy integration with CI

How this works?



Example

```
1  module.exports = {  
2    tags: ['google'],  
3    'Demo test Google' : function (client) {  
4      client  
5        .url('http://www.google.com')    url to navigate first  
6        .waitForElementVisible('body', 1000) wait for element body to be  
7        .assert.title('Google') the title should be 'Google'  
8        .assert.visible('input[type=text]') this input should be visible  
9        .setValue('input[type=text]', 'nightwatch') simulate enter keys.  
10       .waitForElementVisible('button[name=btnG]', 1000) in the input  
11       .click('button[name=btnG]') trigger click event in the button  
12       .pause(1000) element with name="btnG"  
13       .assert.containsText('#main', 'The Night Watch')  
14       .end();  
15     }  
16   };  
17 }
```

Pause 1000 ms

end the test

We are expecting that the div#main should contains the text 'The Night Watch'

Examples

```
1  /* jshint expr: true */
2  module.exports = {
3    tags: ['google'],
4    'Demo test Google' : function (client) {
5      client
6        .url('http://google.no')
7        .pause(1000);
8
9      |
10     client.expect.element('#hplogo').text.to.match(/Norge/).before(1000);
11
12     client.setValue('#lst-ib', 'Norway').pause(500);
13     client.expect.element('#lst-ib').to.have.value.equal('Norway');
14     client.expect.element('#lst-ib').to.be.an('input');
15     client.expect.element('#lst-ib').to.be.not.selected;
16     client.expect.element('#lst-ib').to.be.visible;
17
18     client.end();
19   }
20 };
21
```

```

module.exports = {
  tags: ['google'],
  'Demo test Google' : function (client) {
    client
      .url('http://google.no')
      .pause(1000);

    client.expect.element('#hplogo').text.to.match(/Norge/).before(1000);

    client.setValue('#lst-ib', 'Norway').pause(500);
    client.expect.element('#lst-ib').to.have.value.equal('Norway');
    client.expect.element('#lst-ib').to.be.an('input');
    client.expect.element('#lst-ib').to.be.not.selected;
    client.expect.element('#lst-ib').to.be.visible;

  },
  after : function(client) {
    client.end();
  }
};

```

This chaining expects are using chai library
<http://chaijs.com/api/bdd/>

-Se pueden agregar tags y desp correr solo tags específicos mediante .

`/nightwatch --tag 'google'`

-O bien saltarse los tests con determinado tag: `/nightwatch --skiptags 'google'`

```
describe('New Tests', function() {  
  before(function(client, done) {  
    done();  
  });  
  
  after(function(client, done) {  
    if (client.sessionId) {  
      client  
        .pause(1999)  
        .end(function() {  
          done();  
        });  
    } else {  
      done();  
    }  
  });  
  
  afterEach(function(client, done) {  
    done();  
  });  
  
  beforeEach(function(client, done) {  
    done();  
  });  
  
  it('Case 9U: decimals in inputs', function(client) {
```

This example is using
mocha as test runner

This example is using mocha as test runner

```
it('Case 9U: decimals in inputs', function(client) {  
  client  
    .url('http://google.no')  
    .pause(1000);  
  
  client.expect.element('body').to.be.present;  
  
  client.expect.element('#lst-ib').to.have.css('display');  
  
  client.expect.element('body').to.have.attribute('class').which.contains('vasq')  
  client.expect.element('body').to.have.attribute('class').which.matches(/vasq$/)  
  client.expect.element('body').to.have.attribute('class').before(1000);  
  
  client.expect.element('#lst-ib').to.be.enabled;  
  client.pause();  
});
```

Config nightwach

```
"test_settings" : {  
  "default" : {  
    "launch_url" : "http://localhost",  
    "selenium_port" : 4444,  
    "selenium_host" : "localhost",  
    "silent": true,  
    "screenshots" : {  
      "enabled" : true,  
      "path" : "tests-output/screenshots"  
    },  
    "desiredCapabilities": {  
      "browserName": "chrome",  
      "javascriptEnabled": true,  
      "acceptSslCerts": true  
    }  
  },  
}
```

—

Config nightwach

```
"test_runner" : {  
  "type" : "mocha",  
  "options" : {  
    "ui" : "bdd",  
    "reporter" : "spec"  
  }  
},
```


Nightwatch in action

Only execute
this test

```
1 app = require('../../helpers/helper.js')();
2
3
4 describe('app', function() {
5
6
7     it('Case 2', function(client) {
8         app.case3(client, app.type.preTerm);
9     });
10
11     it.only('Case 3', function(client) {
12         app.case3(client, app.type.postTerm);
13     });
14
15     it('Case 4 flow->yes', function(client) {
16         app.case4(client);
17         client
18             .click(app.popupMessagebuttonValue('yes'))
19             .useXpath()
20             .assert.containsText(app.getDropdownSelected('
21
22     });
23
24 });
```

Config nightwach

```
"src_folders" : ["tests/rounding"],
"output_folder" : "./reports",
"custom_commands_path" : ["./node_modules/nightwatch-commands/commands/", "./custo
"custom_assertions_path" : "./node_modules/nightwatch-commands/commands/",
"page_objects_path" : "./page-objects/",
"globals_path" : "",
"skip_testcases_on_fail": "false",
"end_session_on_fail": "false",

"selenium" : {
  "start_process" : true,
  "server_path" : "./selenium-server.jar",
  "log_path" : "",
  "host" : "127.0.0.1",
  "port" : 4444,
  "cli_args" : {
    "webdriver.chrome.driver" : "./node_modules/chromedriver/bin/chromedriver",
    "webdriver.ie.driver" : ""
  }
}
```


Config nightwach

```
"chrome" : {
  "desiredCapabilities": {
    "browserName": "chrome",
    "javascriptEnabled": true,
    "acceptSslCerts": true,
    "loggingPrefs": { "browser": "ALL" }
  }
},

"firefox" : {
  "desiredCapabilities": {
    "browserName": "firefox"
  }
},

"phantomjs":{
  "desiredCapabilities" : {
    "browserName" : "phantomjs",
    "javascriptEnabled" : true,
    "acceptSslCerts" : true,
    "phantomjs.binary.path" : "./node_modules/phantomjs-prebuilt/bin/phantomjs",
    "phantomjs.cli.args" : ["--version"]
  }
}
```

- `/node_modules/nightwatch/bin/nightwatch`
`-e chrome,firefox,ie,safari,etc`

— custom-commands/waitForElementAndClick.js

```
1 module.exports.command = function(source, dest, callback) {  
2   var self = this;  
3  
4   this  
5   .waitForElementVisible(source, 4000)  
6   .click(source);  
7  
8   return this;  
9 };  
10
```

```
client  
  .waitForElementAndClick('body .custom-class')  
  .useCss()
```

```
module.exports.command = function(source, dest, callback) {  
  var self = this;  
  
  this.moveToElement(source, 10, 10)  
  // .pause(800)  
  // .moveTo(1, 1)  
  .mouseButtonDown(0)  
  .pause(500)  
  .moveToElement(dest, 10, 10)  
  .mouseButtonUp(0);  
  
  return this;  
};
```

— Executing code in the browser using the ".execute()" method

```
.execute(function(data) {  
    try {  
        // statements  
  
        p = document.querySelector("input[type='range']");  
        p.value = 41;  
        event = document.createEvent("HTMLEvents");  
        event.initEvent("change", true, true);  
        p.dispatchEvent(event);  
    } catch(e) {  
        // statements  
        console.log(e);  
    }  
    return true;  
}, [], function(result) {  
    });
```

Continuous integration

Continuous integration

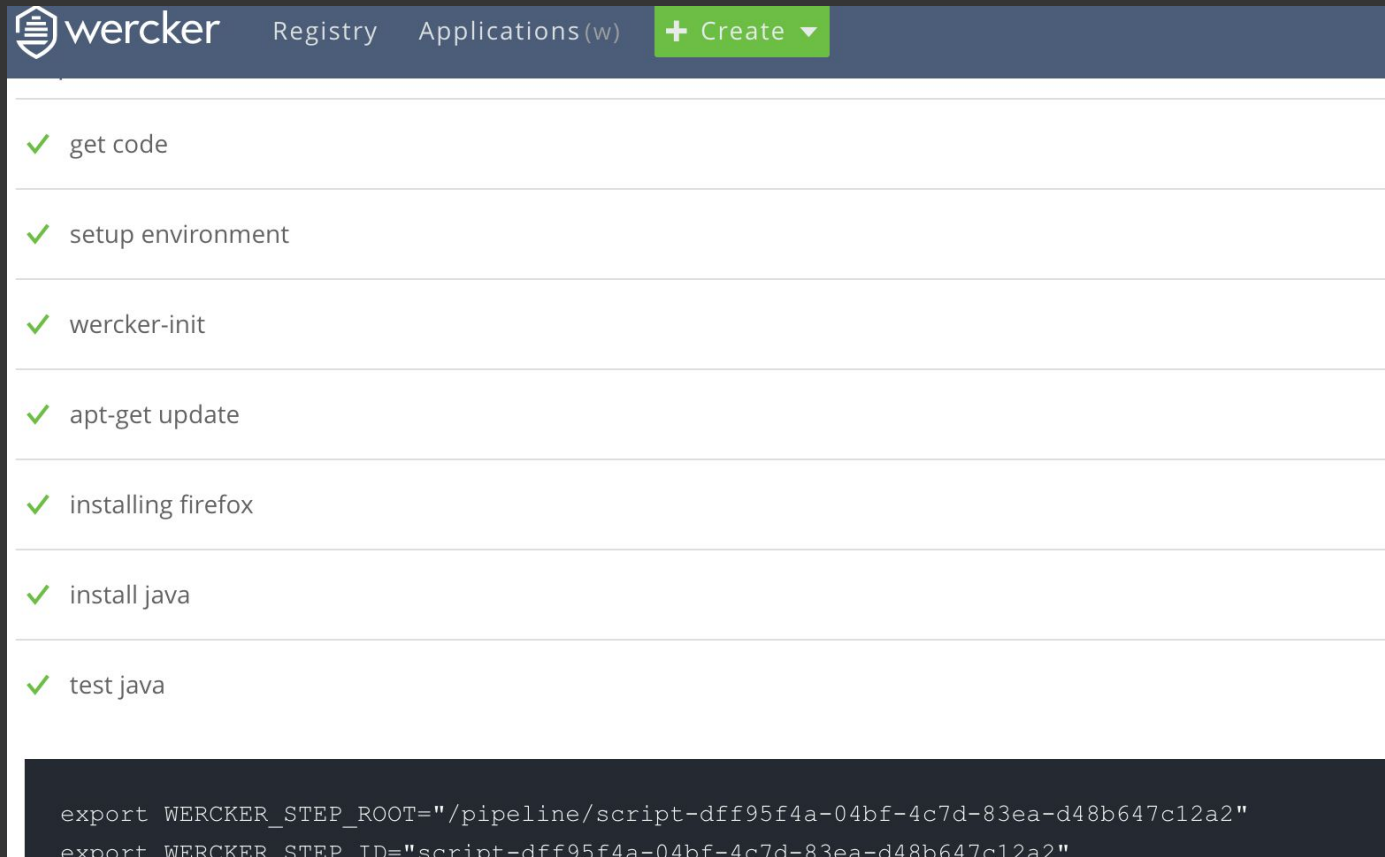
The screenshot displays a list of build attempts in a CI tool. Each entry includes a status icon (blue circle for in progress, red X for failed, green checkmark for success), a build ID, a branch name, and a commit hash. The first build is in progress, indicated by a blue circle and the text 'Building...'. The second and third builds are failed, indicated by a red X and the text 'error: I left .pause(0) pause forever'. The fourth build is failed, indicated by a red X and the text 'error: a test is failing.'. The fifth build is successful, indicated by a green checkmark and the text 'successful build'.

Status	Build ID	Branch	Commit Hash	Message
In Progress	#d5d7ba1...	master	case 60 filed	Building...
Failed	#05443b7...	master	Merge remote-tracking branch 'origin/master'	error: I left .pause(0) pause forever
Failed	#886d023...	master	Merge remote-tracking branch 'origin/master'	error: a test is failing.
Failed	#96a970f...	master	testing CI.	error: a test is failing.
Successful	#cd15933...	master	58.59 done	successful build

This example is with the a CI tool call wercker(is in the cloud) similar tools are:

Thoughworks GO, Travis CI, Jenkins, etc

Continuous integration



The screenshot displays the Wercker web interface. At the top, there is a navigation bar with the Wercker logo, links for 'Registry' and 'Applications (w)', and a green '+ Create' button. Below the navigation bar, a list of pipeline steps is shown, each with a green checkmark indicating success. The steps are: 'get code', 'setup environment', 'wercker-init', 'apt-get update', 'installing firefox', 'install java', and 'test java'. At the bottom of the interface, a dark blue box contains environment variables for the current step.

wercker Registry Applications (w) + Create ▼

- ✓ get code
- ✓ setup environment
- ✓ wercker-init
- ✓ apt-get update
- ✓ installing firefox
- ✓ install java
- ✓ test java

```
export WERCKER_STEP_ROOT="/pipeline/script-dff95f4a-04bf-4c7d-83ea-d48b647c12a2"
export WERCKER_STEP_ID="script-dff95f4a-04bf-4c7d-83ea-d48b647c12a2"
```



```
box: node
build:
  steps:
    - script:
        name: installing firefox
        code: |-
            tar xvjf firefox-45.0.tar.bz2
            sudo ln -s /usr/local/firefox/firefox /usr/bin/firefox
    - script:
        name: install java
        code: |-
            sudo apt-get --assume-yes install default-jre
            type java || { echo "I expected java to be available after oracle-java
    - script:
        name: start virtual display xvfb
        code: |-
            xdpinfo -display :0 >/dev/null 2>&1 && echo "In use" || echo "Free"
    - npm-install
    - script:
        name: set NODE_ENV == production
        code: |-
            export NODE_ENV=testing
    - script:
        name: install nightwatch global
        code: |-
            npm install -g nightwatch
    - script:
        name: running tests
        code: |-
            nightwatch -e firefox
  after-steps:
    - slack-notifier:
```

Continuous integration
Wercker config.



Next Steps:

- Add more tests to widgets
- Migrate Mocha tests in Poe to Nightwatch native test runner to have more feedback about assertions
- Every member of the team must write a quick test of a familiar widget
- Run accessibility tests and html validation tests

Preguntas/Questions?

More info in:

- nightwatchjs.org/
- groups.google.com/forum/#!forum/nightwatchjs
- www.stackoverflow.com



Gracias! / Thanks

mail : davidtorroija@gmail.com

example Repo:

github.com/davidtorroija/nightwatch-example