

What is Nightwatch

nightwatch v1.0.16

 Weekly downloads

155k+

author Jagadeesh Shetty

Nightwatch.js is an automated testing framework for web applications and websites, written in Node.js and using the [W3C WebDriver](#) API (formerly [Selenium WebDriver](#)).

It is a complete End-to-End testing solution which aims to simplify writing automated tests and setting up Continuous Integration. Nightwatch can also be used for writing Node.js unit and integration tests.

The name Nightwatch was inspired by the famous painting The Night Watch by Dutch artist Rembrandt van Rijn. The masterpiece is prominently displayed in the Rijksmuseum, in Amsterdam - The Netherlands.

Overview of WebDriver

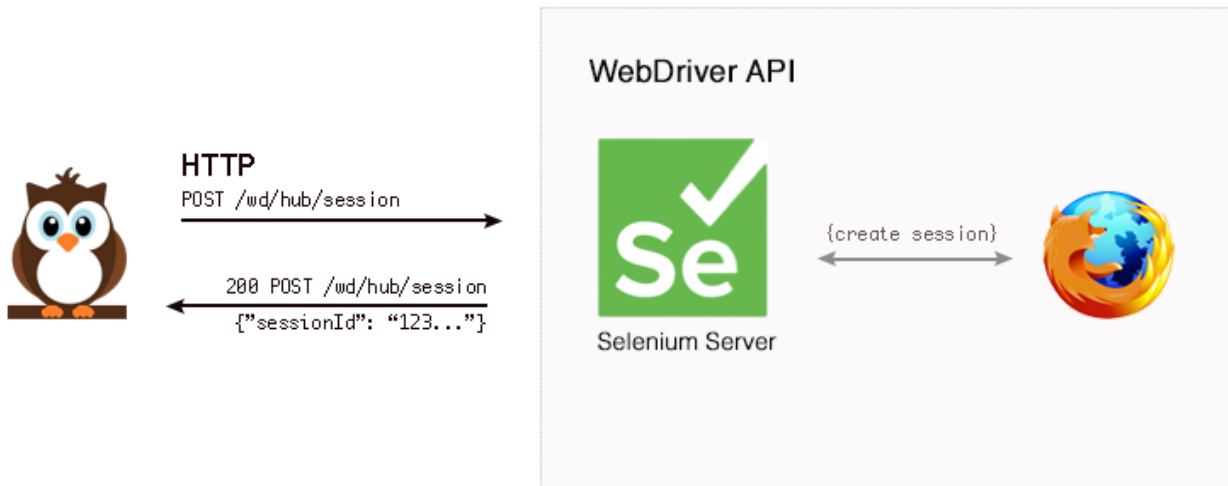
WebDriver is a general purpose library for automating web browsers. It was started as part of the [Selenium](#) project, which is a popular and comprehensive set of tools for browser automation initially written for Java but now with support for most programming languages.

Nightwatch uses the [WebDriver API](#) to perform the browser automation related tasks, like opening windows and clicking links for instance.

WebDriver is now a W3C specification aiming to standardize browser automation. WebDriver is a remote control interface that enables introspection and control of user agents. It provides a platform and a restful HTTP api as a way for web browsers to be remotely controlled.

Theory of Operation

Nightwatch works by communicating over a restful HTTP API with a WebDriver server (such as ChromeDriver or Selenium Server). The protocol is defined by the W3C WebDriver spec, which is derived from [JSON Wire protocol](#). See below for an example workflow for browser initialization.



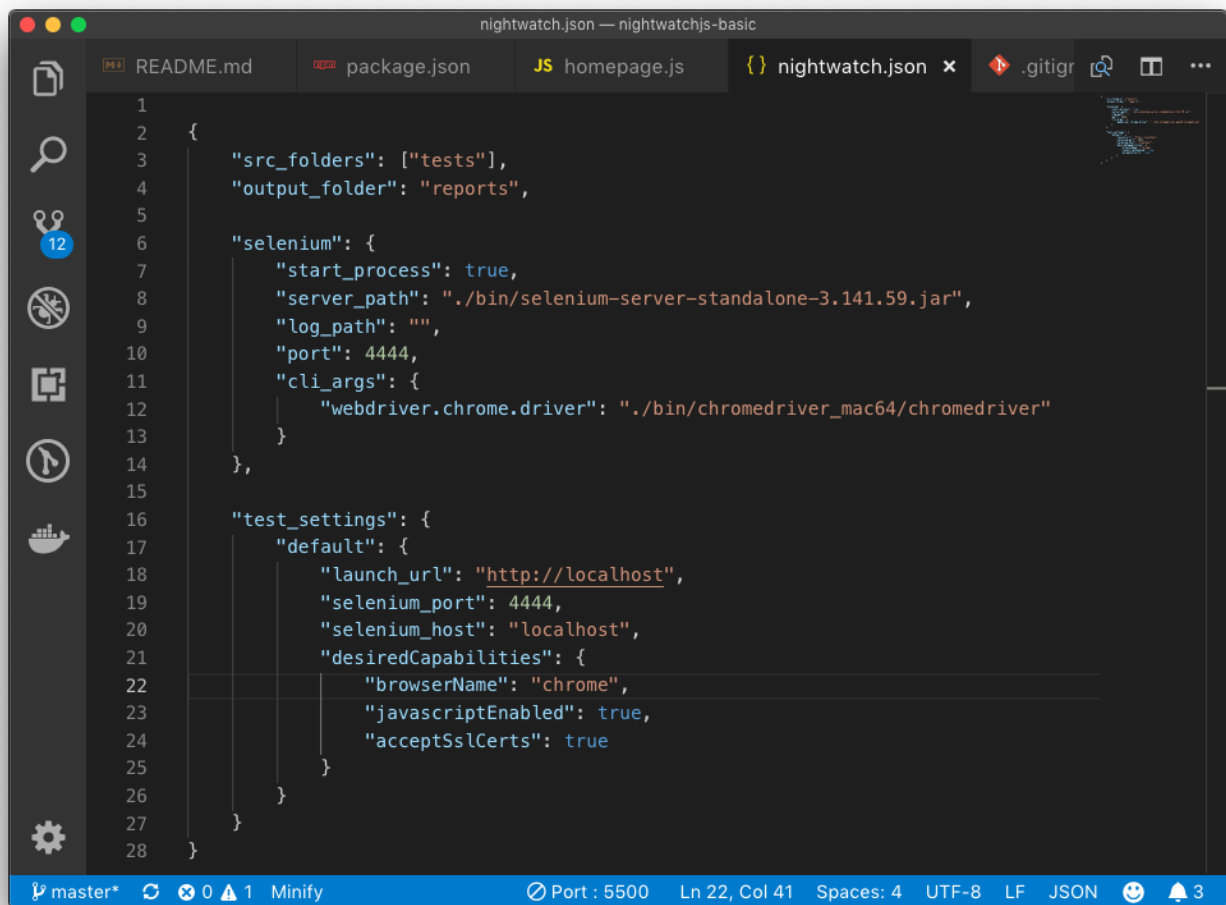
Most of the times, Nightwatch needs to send at least 2 requests to the WebDriver server in order to perform a command or assertion. ...The first one being the request to locate an element given a CSS selector (or Xpath expression) ...Next to perform the actual command/assertion on the given element.

Setup

1. **Node.js** should be installed.
2. **NPM** should be installed.
3. Create a directory with **nightwatchjs-basic** name.
4. Run `npm init` and complete with basic details.
5. Run `npm install nightwatch` to install nightwatch within workspace.
6. Download `selenium` and `chrome` drivers and copy to `bin` directory.
7. Create `nightwatch.json` file within working directory. The nightwatch test runner binary expects a configuration file.
8. Create `tests` directory and create `homepage.js` file.
9. Update `package.json` with test runner for `scripts` value.

Screenshots

nightwatch.conf

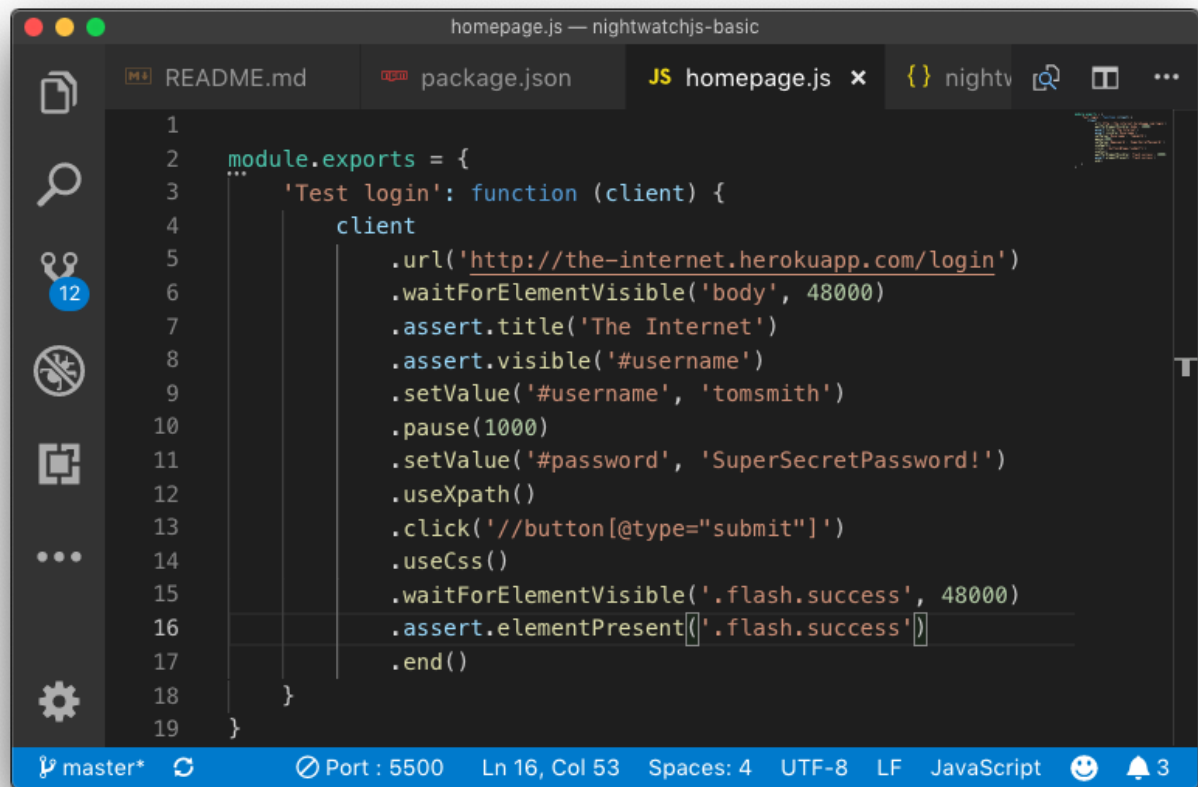


The screenshot shows a code editor window titled "nightwatch.json — nightwatchjs-basic". The editor has several tabs open: "README.md", "package.json", "JS homepage.js", and "nightwatch.json" (which is the active tab). The "nightwatch.json" file contains the following JSON configuration:

```
1 {
2   "src_folders": ["tests"],
3   "output_folder": "reports",
4
5   "selenium": {
6     "start_process": true,
7     "server_path": "./bin/selenium-server-standalone-3.141.59.jar",
8     "log_path": "",
9     "port": 4444,
10    "cli_args": {
11      "webdriver.chrome.driver": "./bin/chromedriver_mac64/chromedriver"
12    }
13  },
14
15  "test_settings": {
16    "default": {
17      "launch_url": "http://localhost",
18      "selenium_port": 4444,
19      "selenium_host": "localhost",
20      "desiredCapabilities": {
21        "browserName": "chrome",
22        "javascriptEnabled": true,
23        "acceptSslCerts": true
24      }
25    }
26  }
27 }
28 }
```

The editor's status bar at the bottom shows "master*", "0" errors, "1" warning, "Minify", "Port : 5500", "Ln 22, Col 41", "Spaces: 4", "UTF-8", "LF", "JSON", and "3" notifications.

tests/homepage.js

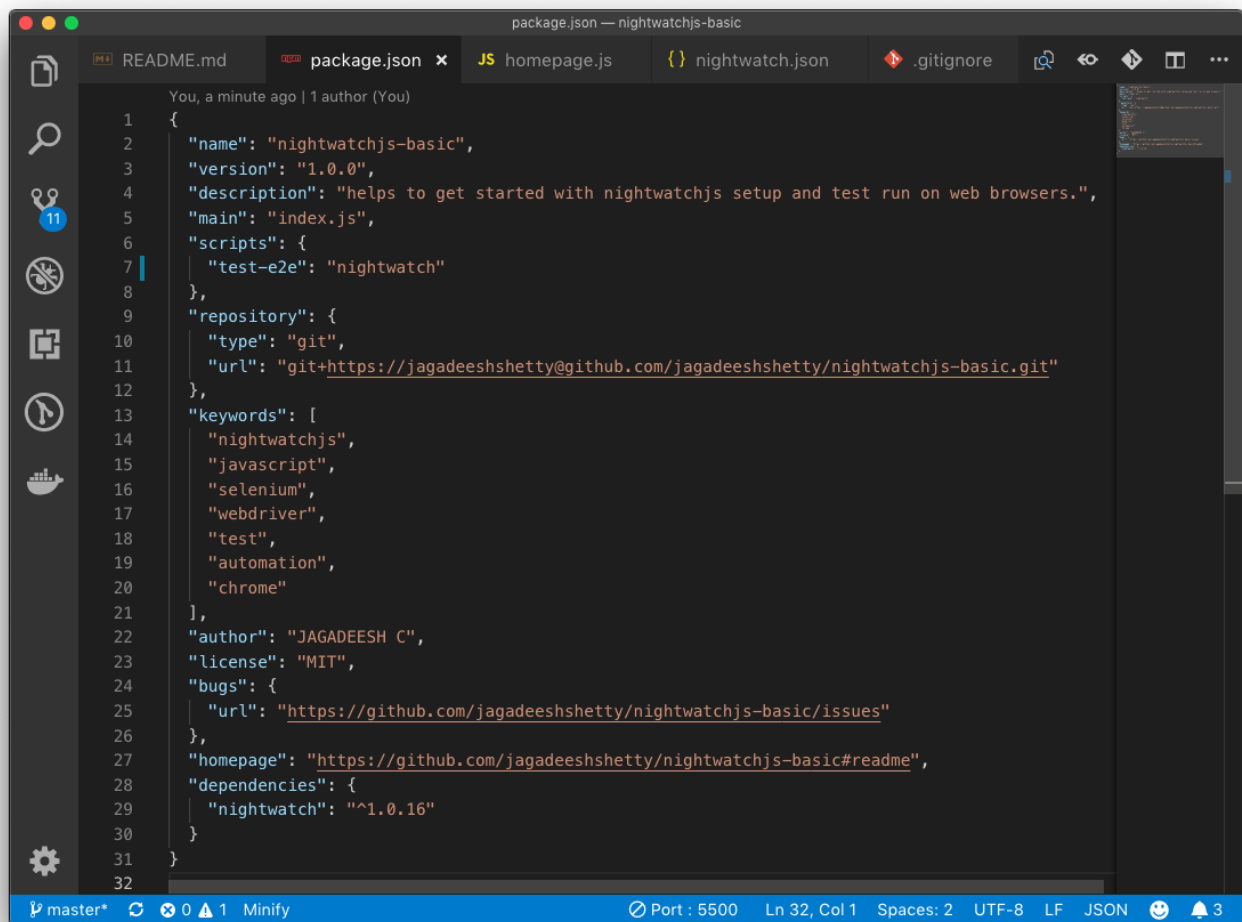


The screenshot shows a code editor window titled "homepage.js — nightwatchjs-basic". The editor has tabs for "README.md", "package.json", and "JS homepage.js". The "JS homepage.js" tab is active, displaying a Nightwatch.js test script. The script is as follows:

```
1
2 module.exports = {
3   ...
4   'Test login': function (client) {
5     client
6       .url('http://the-internet.herokuapp.com/login')
7       .waitForElementVisible('body', 48000)
8       .assert.title('The Internet')
9       .assert.visible('#username')
10      .setValue('#username', 'tomsmith')
11      .pause(1000)
12      .setValue('#password', 'SuperSecretPassword!')
13      .useXpath()
14      .click('//button[@type="submit"]')
15      .useCss()
16      .waitForElementVisible('.flash.success', 48000)
17      .assert.elementPresent($('.flash.success'))
18      .end()
19   }
20 }
```

The status bar at the bottom indicates the current branch is "master*", the port is "5500", and the cursor is at "Ln 16, Col 53". Other settings shown are "Spaces: 4", "UTF-8", "LF", and "JavaScript". There are also icons for a smiley face and a bell with the number "3".

package.json

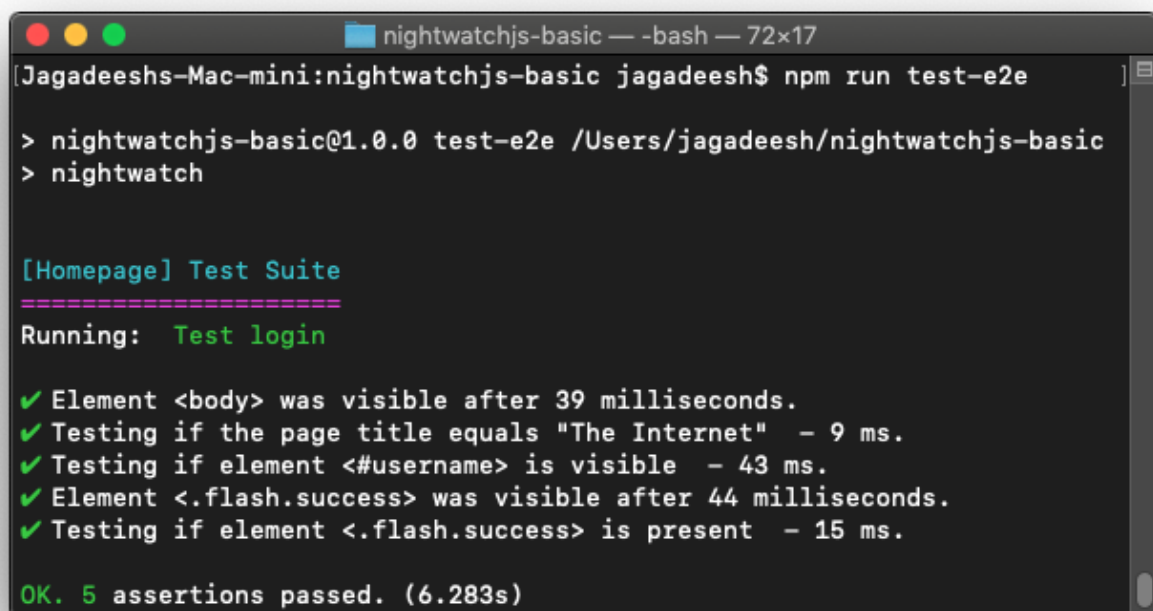


The screenshot shows a code editor with the 'package.json' file open. The file contains the following JSON structure:

```
{
  "name": "nightwatchjs-basic",
  "version": "1.0.0",
  "description": "helps to get started with nightwatchjs setup and test run on web browsers.",
  "main": "index.js",
  "scripts": {
    "test-e2e": "nightwatch"
  },
  "repository": {
    "type": "git",
    "url": "git+https://jagadeeshshetty@github.com/jagadeeshshetty/nightwatchjs-basic.git"
  },
  "keywords": [
    "nightwatchjs",
    "javascript",
    "selenium",
    "webdriver",
    "test",
    "automation",
    "chrome"
  ],
  "author": "JAGADEESH C",
  "license": "MIT",
  "bugs": {
    "url": "https://github.com/jagadeeshshetty/nightwatchjs-basic/issues"
  },
  "homepage": "https://github.com/jagadeeshshetty/nightwatchjs-basic#readme",
  "dependencies": {
    "nightwatch": "^1.0.16"
  }
}
```

The editor interface includes a sidebar with icons for file explorer, search, and other tools. The status bar at the bottom shows 'master', '0', '1', 'Minify', 'Port: 5500', 'Ln 32, Col 1', 'Spaces: 2', 'UTF-8', 'LF', 'JSON', and a notification bell.

Run



The screenshot shows a terminal window with the following output:

```
Jagadeeshs-Mac-mini:nightwatchjs-basic jagadeesh$ npm run test-e2e

> nightwatchjs-basic@1.0.0 test-e2e /Users/jagadeesh/nightwatchjs-basic
> nightwatch

[Homepage] Test Suite
=====
Running: Test login

✓ Element <body> was visible after 39 milliseconds.
✓ Testing if the page title equals "The Internet" - 9 ms.
✓ Testing if element <#username> is visible - 43 ms.
✓ Element <.flash.success> was visible after 44 milliseconds.
✓ Testing if element <.flash.success> is present - 15 ms.

OK. 5 assertions passed. (6.283s)
```

Reference

[Official Nightwatch](#)