Automated testing with Headless Chrome

By Eric Bidelman

Engineer @ Google working on web tooling: Headless Chrome, Puppeteer, Lighthouse

If you want to run automated tests using Headless Chrome, look no further! This article will get you all set up using Karma as a runner and Mocha+Chai for authoring tests.

What are these things?

Karma, Mocha, Chai, Headless Chrome, oh my!

Karma is a testing harness that works with any of the the most popular testing frameworks (Jasmine, Mocha, QUnit).

Chai is an assertion library that works with Node and in the browser. We need the latter.

Headless Chrome is a way to run the Chrome browser in a headless environment without the full browser UI. One of the benefits of using Headless Chrome (as opposed to testing directly in Node) is that your JavaScript tests will be executed in the same environment as users of your site. Headless Chrome gives you a real browser context without the memory overhead of running a full version of Chrome.

Setup

Installation

Install Karma, the relevant, plugins, and the test runners using `yarn`:

```
yarn add --dev karma karma-chrome-launcher karma-mocha karma-chai
yarn add --dev mocha chai
```

or use `npm`:

```
npm i --save-dev karma karma-chrome-launcher karma-mocha karma-chai
npm i --save-dev mocha chai
```

I'm using Mocha and Chai in this post, but if you're not a fan, choose your favorite assertion library that works in the browser.

Configure Karma

Create a `karma.config.js` file that uses the `ChromeHeadless` launcher.

karma.conf.js

```
module.exports = function(config) {
  config.set({
    frameworks: ['mocha', 'chai'],
    files: ['test/**/*.js'],
    reporters: ['progress'],
    port: 9876,  // karma web server port
    colors: true,
    logLevel: config.LOG_INFO,
    browsers: ['ChromeHeadless'],
    autoWatch: false,
    // singleRun: false, // Karma captures browsers, runs the tests and exits
    concurrency: Infinity
  })
}
```

Note: Run `./node_modules/karma/bin/ init karma.conf.js` to generate the Karma configuration file.

Write a test

Create a test in `/test/test.js`.

/test/test.js

```
describe('Array', () => {
  describe('#indexOf()', () => {
    it('should return -1 when the value is not present', () => {
      assert.equal(-1, [1,2,3].indexOf(4));
    });
  });
});
```

Run your tests

Add a `test` script in `package.json` that runs Karma with our settings.

package.json

```
"scripts": {
  "test": "karma start --single-run --browsers ChromeHeadless karma.conf.js"
}
```

When you run your tests (`yarn test`), Headless Chrome should fire up and output the results to the terminal:

```
yarn test v0.24.6
$ karma start --single-run --browsers MyHeadlessChrome karma.conf.js
13 06 2017 10:49:44.947:INFO [karma]: Karma v1.7.0 server started at http://0.0.0.0:9876/
13 06 2017 10:49:44.950:INFO [launcher]: Launching browser MyHeadlessChrome with unlimited concurrency
13 06 2017 10:49:44.957:INFO [launcher]: Starting browser ChromeHeadless
13 06 2017 10:49:45.269:INFO [HeadlessChrome 0.0.0 (Mac OS X 10.12.4)]: Connected on socket xA4bkPZP9uYEWoMRAAAA with id 86428182
HeadlessChrome 0.0.0 (Mac OS X 10.12.4): Executed 1 of 1 SUCCESS (0.008 secs / 0.001 secs)
✨ Done in 1.60s.
```

Creating your own Headless Chrome launcher

The `ChromeHeadless` launcher is great because it works out of the box for testing on Headless Chrome. It includes the appropriate Chrome flags for you and launches a remote debugging version of Chrome on port `9222`.

However, sometimes you may want to pass custom flags to Chrome or change the remote debugging port the launcher uses. To do that, create a `customLaunchers` field that extends the base `ChromeHeadless` launcher:

karma.conf.js

```
module.exports = function(config) {
  ...

  config.set({
    browsers: ['Chrome', 'ChromeHeadless', 'MyHeadlessChrome'],

    customLaunchers: {
      MyHeadlessChrome: {
        base: 'ChromeHeadless',
        flags: ['--disable-translate', '--disable-extensions', '--remote-debugging-p
      }
    },
  }
};
```

Running it all on Travis CI

Configuring Karma to run your tests in Headless Chrome is the hard part. Continuous integration in Travis is just a few lines away!

To run your tests in Travis, use `dist: trusty` and install the Chrome stable addon:

.travis.yml

```
language: node_js
node_js:
  - "7"
dist: trusty # needs Ubuntu Trusty
# Note: if you switch to sudo: false, you'll need to launch chrome with --no-sandbox
# See https://github.com/travis-ci/travis-ci/issues/8836
sudo: required
addons:
  chrome: stable # have Travis install chrome stable.
cache:
  yarn: true
  directories:
    - node_modules
install:
  - yarn
script:
  - yarn test
```

Note: check out the example repo for reference.

Last updated July 2, 2018.