



Web Development

COMP 431 / COMP 531

Lecture 14: End-to-End Testing

Instructor: Mack Joyner
Department of Computer Science, Rice University

mjoyner@rice.edu

<http://www.clear.rice.edu/comp431>

Frontend Recap

- HTML and HTML5, Storage, Canvas
- JavaScript and Scope
- Forms, CSS, Events
- jQuery, AJAX, and fetch
- Modern JS
- MVC, Angular, Testing

Quiz 2

Due **Thursday** 10/19

Homework Assignment 5
(Front-End App)

Due Thursday 10/26

HW 5 Front-End Web Application

- TDD = *Write tests first!*
- Implement:
 - User login / logout – routing from landing to main to profile views
 - Error message for invalid users
 - Status headline update (persistent)
 - ~~Filter the posts by author/body (not id)~~ [already done]
 - Add a post (persistent)
 - Comment on a post
 - Update profile page to match registered logged in user (not hardcoded)

istanbul public



Yet another JS code coverage tool that computes statement, line, function and branch coverage with module loader hooks to transparently add coverage when running tests. Supports all JS coverage use cases including unit tests, server side functional tests

Istanbul - a JS code coverage tool written in JS

build

passing

dependencies

out-of-date

coverage

98%



bitHound

90



npm install istanbul

14 dependencies version 0.4.5
405 dependents updated a month ago
2,861,339 downloads in the last month

- Instruments code
- Generates coverage report

Two Approaches

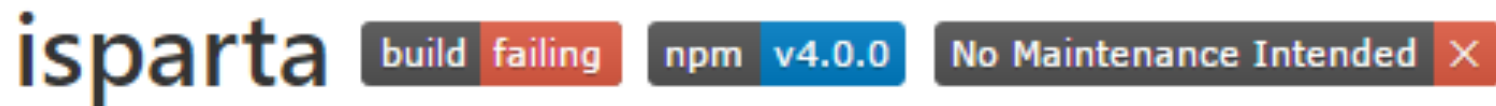
Istanbul with mocha

- ES6 requires isparta

Karma with mocha

- Karma is a test runner
- Launches a browser

- isparta is no longer supported



npm scripts

```
"karma": "karma start --single-run",  
"karma:watch": "karma start",
```



- Karma is a test runner used to run tests in a browser environment

npm run karma

```
27 09 2015 21:44:21.127:WARN [karma]: No captured browser, open http://localhost:9876/
27 09 2015 21:44:21.158:INFO [karma]: Karma v0.13.10 server started at http://localhost:9876/
27 09 2015 21:44:21.221:INFO [launcher]: Starting browser Chrome
27 09 2015 21:44:29.937:INFO [Chrome 45.0.2454 (Windows 8.1 0.0.0)]: Connected on socket Z_nU9i_rt8KCs7
QCAAAA with id 86007040
Chrome 45.0.2454 (Windows 8.1 0.0.0): Executed 5 of 5 SUCCESS (0.164 secs / 0.121 secs)
```

- Opened a (captured) Browser
- Use the browser to execute code
 - Runs all the tests
- Collected results from the browser
- Presents the test results in the terminal



```
1 //*****//
2 // Karma Test Runner //
3 //*****//
4
5 var babelrc = JSON.parse(require('fs').readFileSync('.babelrc').toString())
6
7 // We use webpack to resolve import/require statements
8 var webpackConfig = require('./webpack.config.js')
9 webpackConfig.entry = {}
10 // inline the source map instead of a separate file
11 webpackConfig.devtool = 'inline-source-map'
12 // instrumentation for coverage
13 if (!webpackConfig.module.preLoaders) webpackConfig.module.preLoaders = []
14 webpackConfig.module.preLoaders.push({
15   test: /\.jsx?$/,
16   include: /src/,
17   exclude: /(node_modules)/,
18   loader: 'babel-istanbul',
19   query: {
20     cacheDirectory: true
21   })
22 webpackConfig.resolve = {
23   alias: {
24     'isomorphic-fetch': 'mock-fetch',
25   }
26 }
27 webpackConfig.externals = {
```

karma.conf.js


```
26 }
27 webpackConfig.externals = {
28   'jsdom': 'window',
29   'mockery': 'window',
30 }
31
32 module.exports = function(config) {
33   config.set({
34     autoWatch: true,
35     singleRun: false,
36     browsers: ['Chrome'],
37     frameworks: ['mocha'],
38     logLevel: config.LOG_INFO,
39     files: [ 'tests.webpack.js' ],
40     preprocessors: {
41       'tests.webpack.js': ['webpack', 'sourcemap']
42     },
43     webpack: webpackConfig,
44     webpackMiddleware: { noInfo: true },
45     coverageReporter: {
46       reporters: [
47         { type: 'html', subdir: 'html' },
48         { type: 'lcovonly', subdir: '.' },
49       ],
50     },
51     reporters: ['progress', 'coverage'],
52   })
53 }
```

karma.conf.js

/

50.56% Statements 45/89 75% Branches 12/16 41.38% Functions 12/29 45.68% Lines 37/81 1 branch Ignored



File ▲		Statements ▾		Branches ▾		Functions ▾		Lines ▾	
src/	<div><div></div></div>	50.56%	45/89	75%	12/16	41.38%	12/29	45.68%	37/81

all files src/

50.56% Statements 45/89 75% Branches 12/16 41.38% Functions 12/29 45.68% Lines 37/81 1 branch Ignored



File ▲		Statements ▾		Branches ▾		Functions ▾		Lines ▾	
dummy.js	<div><div></div></div>	85.71%	36/42	66.67%	8/12	73.33%	11/15	83.33%	30/36
dummy.spec.js	<div><div></div></div>	17.07%	7/41	100%	4/4	8.33%	1/12	12.82%	5/39
index.js	<div><div></div></div>	33.33%	2/6	100%	0/0	0%	0/2	33.33%	2/6

```
1  1x  const url = 'https://webdev-dummy.herokuapp.com'
2
3  1x  const resource = (method, endpoint, payload) => {
4  3x    const options = {
5        method,
6        credentials: 'include',
7        headers: {
8          'Content-Type': 'application/json'
9        }
10     }
11  3x    if (payload) options.body = JSON.stringify(payload)
12
13  3x    return fetch(`${url}/${endpoint}`, options)
14         .then(r => {
15  3x      E if (r.status === 200) {
16  3x        return (r.headers.get('Content-Type').indexOf('json') > 0 ? r.json() : r.text())
17      } else {
18        // useful for debugging, but remove in production
19        console.error(`${method} ${endpoint} ${r.statusText}`)
20        throw new Error(r.statusText)
21      }
22    })
23 }
```

End-to-End Testing

- Karma launched Chrome
- Karma can launch other browsers too!
 - and run all in parallel
- But Karma executes “unit tests”
- Integration, or End-to-End tests, require a full browsing experience
 - We want a real browser and drive that browser, imitating a user



You'll need to download the **chromedriver**

The driver needs to be in your path to execute.

```
npm install -g webdriver-manager
```

```
webdriver-manager update --chrome --standalone
```

which chromedriver

```
npm run build
```

```
npm start &
```

```
npm run e2e
```

```
Test Dummy Server Example Page
```

```
✓ should log in as the test user (899ms)
```

```
✓ Update the headline and verify the change (14002ms)
```

```
shutdown
```

```
✓ now (339ms)
```

```
3 passing (25s)
```

End-to-End Test Report

```
npm run e2e:xunit
```

```
> starter@1.0.0 e2e:xunit
```

```
> npm run e2e:base -- --reporter xunit > e2e/results.xml
```

```
cat e2e/results.xml
```

```
> starter@1.0.0 e2e:base
```

```
> mocha --opts e2e/mocha.opts e2e/*.spec.js "--reporter" "xunit"
```

```
<testsuite name="Mocha Tests" tests="3" failures="0" errors="0" skipped="0" timestamp="Sun  
631">
```

```
<testcase classname="Test Dummy Server Example Page" name="should log in as the test user"
```

```
<testcase classname="Test Dummy Server Example Page" name="Update the headline and verify
```

```
<testcase classname="shutdown" name="now" time="0.514"/>
```

```
</testsuite>
```

Headless Browsing

“A **headless browser** is a web browser without a graphical user interface” – Wikipedia

What did we just do with Selenium?

We controlled a browser programmatically!

... why do we need the GUI?



Karma and **Selenium** are set to use **Chrome** but we can have them use **PhantomJS** for headless testing instead.

Headless Browsing with Casper

```
npm install -g casperjs@1.1.0-beta3
```

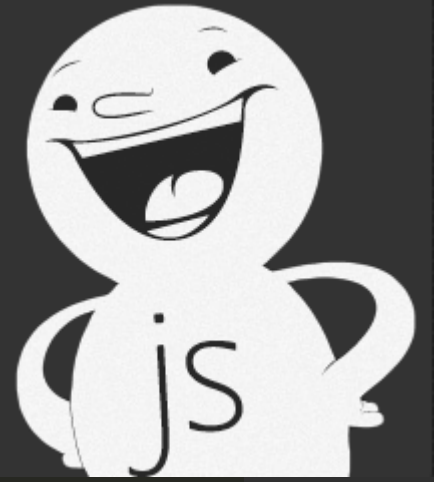
```
var casper = require('casper').create({  
  viewportSize: {width: 1024, height: 768}  
})
```

```
casper.start('https://www.google.com', function() {  
  this.echo(this.getTitle())  
})
```

```
casper.run()
```

```
# casperjs ./haunt.js
```

```
Google
```



Assignments: *a look ahead...*

HW5: Frontend (due 10/26)

- Functional Angular
- Unit tests with Coverage
- Login / Logout
- Articles and search
- Status headline
- Followers

HW6: Draft Backend (11/9)

- Finalize Frontend
 - Upload images
 - Edit profile
 - **End-to-End Tests**
- Draft Backend
 - Connect to server