

O'REILLY®

# Fluent

THE WEB PLATFORM

[fluentconf.com](http://fluentconf.com)

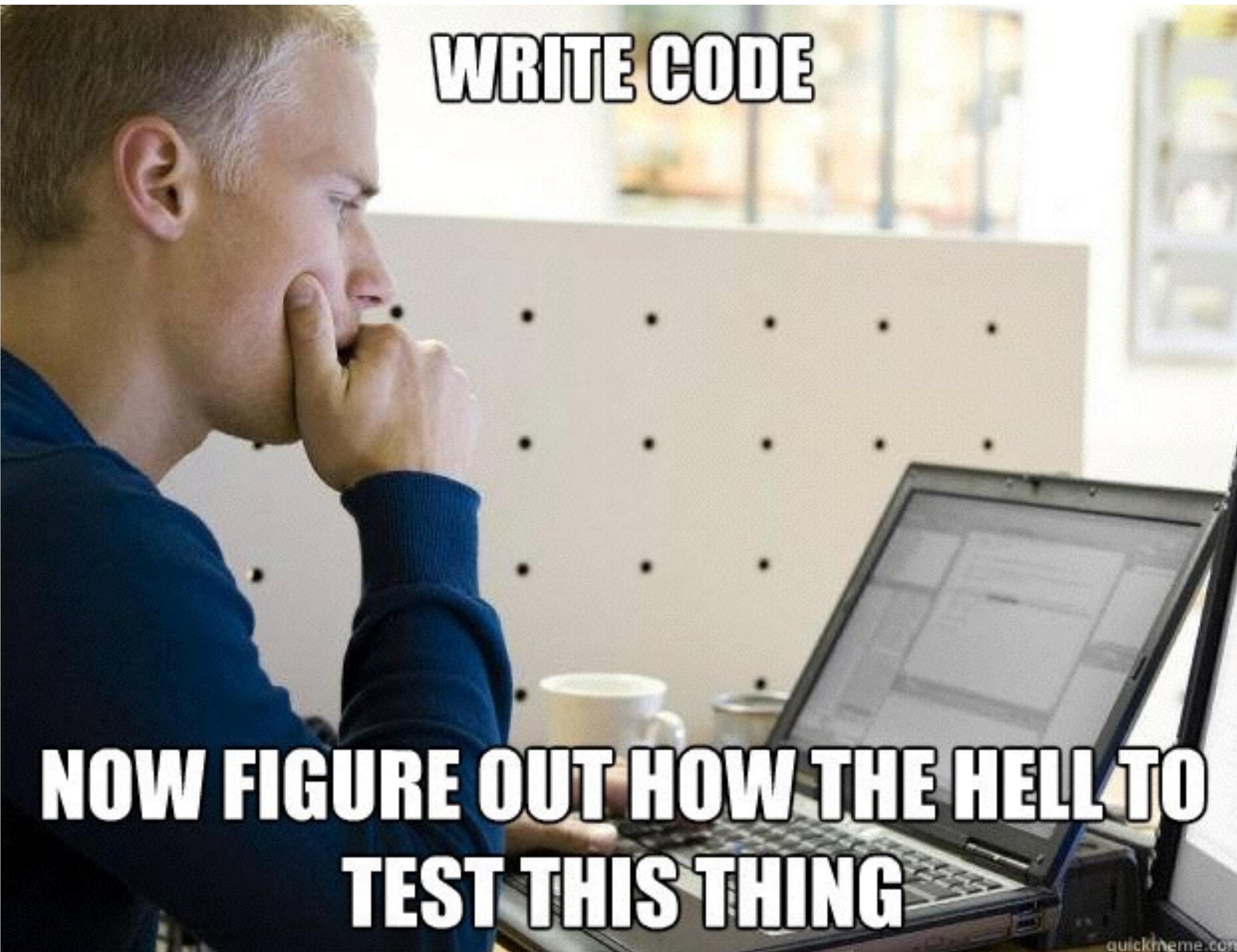
#fluentconf

@JSArtisan

## End-to-end testing for progressively enhanced sites

Brian Belhumeur  
Front End Developer,  
[craigslist.org](http://craigslist.org)

Twitter: @JSArtisan



**WRITE CODE**

**NOW FIGURE OUT HOW THE HELL TO  
TEST THIS THING**

quickmeme.com

Source: <http://www.quickmeme.com/meme/355b3t>



Source: <http://i.imgur.com/U6GJVg2.jpg>



**Slide Deck!**



**Code  
samples!**

Twitter:  
**@JSArtisan**

# #ProgressivelyAwesome

#fluentconf

@JSArtisan

O'REILLY®  
Fluent

Source: <http://dab1nmslvvntp.cloudfront.net/wp-content/uploads/2015/07/1438315888ES6-Support-Table.png>



Source: [http://i.telegraph.co.uk/multimedia/archive/03478/bttf2\\_3478174a.jpg](http://i.telegraph.co.uk/multimedia/archive/03478/bttf2_3478174a.jpg)

Source: <https://d.ibtimes.co.uk/en/full/1416993/marty-mcfly-uses-mattel-hoverboard-escape-thugs-2015.jpg>



Application: HTML + CSS + JavaScript + browser

Progressive enhancement: Basic site + feature tests + selective JS upgrades

End-to-end tests: testing entire finished product as a whole

SF bay area antiques - craig... +

https://sfbay.craigslist.org/search/ata

CL SF bay area > all SF bay area > for sale > antiques

2 favorites [ account ] post

antiques

all owner dealer

search titles only  
 has image  
 posted today  
 search nearby areas

MILES FROM ZIP  
miles from zip

PRICE  
min max

+ condition

reset update search

• safety tips  
• prohibited items  
• product recalls  
• avoiding scams

search antiques

list thumb gallery map

<< < prev 1 to 100 of 2500 next >

newest \$\$\$ \$\$\$

★ Feb 29 EXCLUSIVE ANTIQUE ART HISTORIC RUG \$2500 (SOMA / south beach) pic map ✖

★ Feb 29 Antique Large Bronze Hand Mirror \$75 (dublin / pleasanton / livermore) pic map ✖

★ Feb 29 Rococo Style Bench's \$295 (willow glen / cambrian) pic map ✖

★ Feb 29 Vintage Chinese Hand-carved jade Peacock \$65 (sunnyvale) pic map ✖

★ Feb 29 Large Old Chinese Scroll Hand Painting \$120 (sunnyvale) pic map ✖ ←

★ Feb 29 A set of 3 Chinese Snuff Bottles \$50 (sunnyvale) pic map ✖

★ Feb 29 Vintage Chinese Beautiful jade Bird \$75 (sunnyvale) pic map ✖

★ Feb 29 Beautiful Vintage Chinese Jade \$30 (dublin / pleasanton / livermore) pic map ✖

★ Feb 29 Two Vintage Old Jade Thumb Rings \$45 (sunnyvale) pic map ✖

★ Feb 29 Antique Sofa and Chair Ornate wood carvings \$2000 (south san francisco) pic map ✖

★ Feb 29 Antique Ceramic Capodimonte lamps 1900's \$800 (daly city) pic map ✖

★ Feb 29 Ridgeway Grandfather Clock w/ papers and original receipt \$700 (daly city) pic map ✖

★ Feb 29 Antique Living Room Set \$4000 (daly city) pic map ✖

**favorites**

SF bay area antiques - craig... +

https://sfbay.craigslist.org/search/ata

CL SF bay area > all SF bay area > for sale > antiques go [ account ] post

antiques

search antiques

<< < prev 1 to 100 of 2500 next >

newest \$\$\$ \$\$\$

search titles only  
 has image  
 posted today  
 search nearby areas

MILES FROM ZIP  
miles from zip

PRICE  
min max

+ condition  
 new  
 like new  
 excellent  
 good  
 fair  
 salvage

reset update search

• safety tips  
• prohibited items  
• product recalls  
• avoiding scams

Feb 29 EXCLUSIVE ANTIQUE ART HISTORIC RUG \$2500 (SOMA / south beach) pic map

Feb 29 Antique Large Bronze Hand Mirror \$75 (dublin / pleasanton / livermore) pic map

Feb 29 Rococo Style Bench's \$295 (willow glen / cambrian) pic map

Feb 29 Vintage Chinese Hand-carved jade Peacock \$65 (sunnyvale) pic map

Feb 29 Large Old Chinese Scroll Hand Painting \$120 (sunnyvale) pic map

Feb 29 A set of 3 Chinese Snuff Bottles \$50 (sunnyvale) pic map

Feb 29 Vintage Chinese Beautiful jade Bird \$75 (sunnyvale) pic map

Feb 29 Beautiful Vintage Chinese Jade \$30 (dublin / pleasanton / livermore) pic map

Feb 29 Two Vintage Old Jade Thumb Rings \$45 (sunnyvale) pic map

Feb 29 Antique Sofa and Chair Ornate wood carvings \$2000 (south san francisco) pic map

Feb 29 Antique Ceramic Capodimonte lamps 1900's \$800 (daly city) pic map

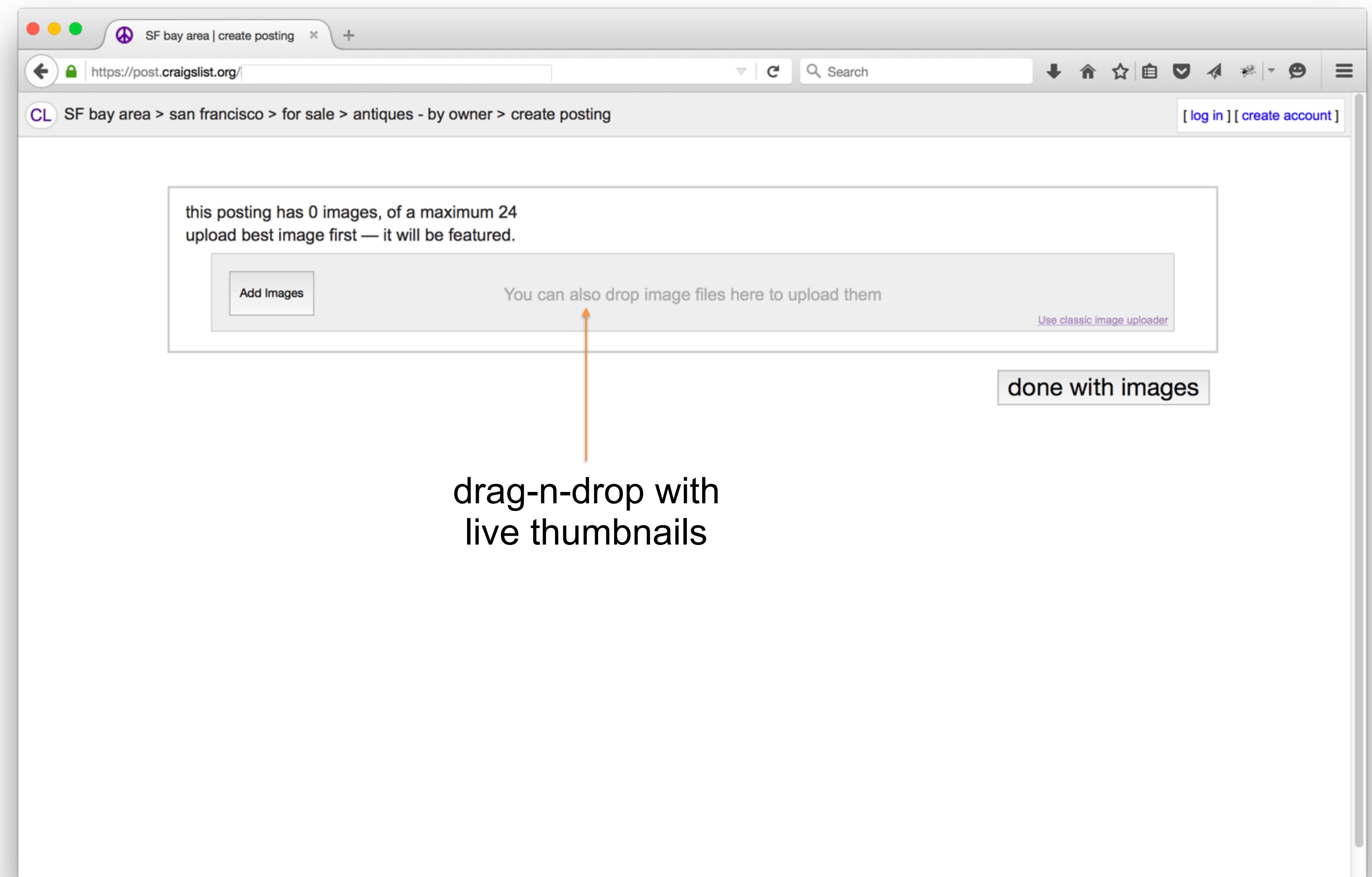
Feb 29 Ridgeway Grandfather Clock w/ papers and original receipt \$700 (daly city) pic map

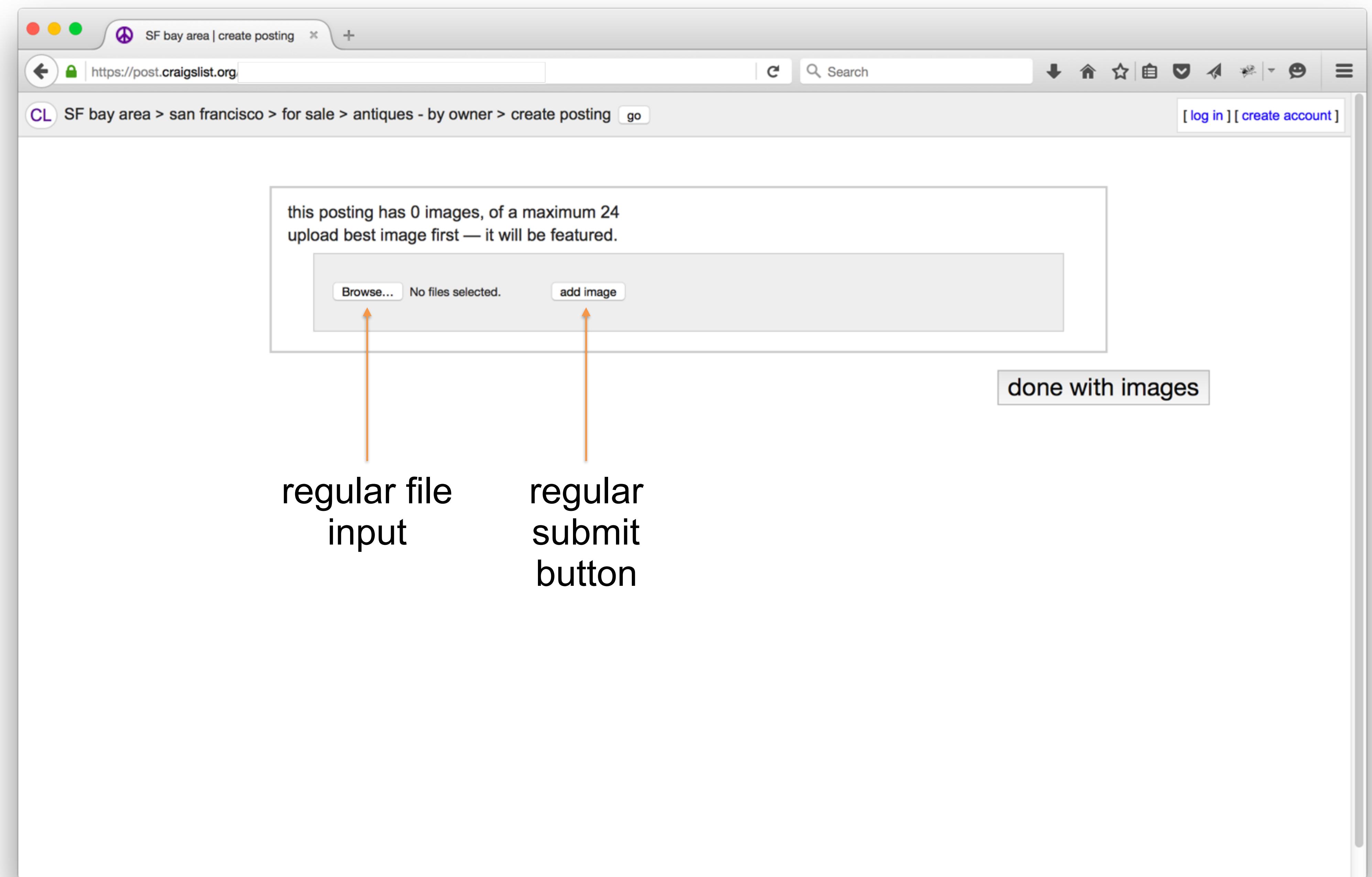
Feb 29 Antique Living Room Set \$4000 (daly city) pic map

Feb 29 Beautiful Vintage Chinese Jade \$30 (dublin / pleasanton / livermore) pic map

Feb 29 Vintage Chinese Beautiful jade Bird \$75 (sunnyvale) pic map

Feb 29 English wash stand \$650 (dublin / pleasanton / livermore) pic





Application: HTML + CSS + JavaScript + browser

Progressive enhancement: Basic site + feature tests + selective JS upgrades

End-to-end tests: testing entire finished product as a whole



# Nightwatch.js

Browser automated testing done easy.

```
google.js x
1 module.exports = {
2   tags: ['google'],
3   'Demo test Google' : function (client) {
4     client
5       .url('http://www.google.com')
6       .waitForElementVisible('body', 1000)
7       .assert.title('Google')
8       .assert.visible('input[type=text]')
9       .setValue('input[type=text]', 'nightwatch')
10      .waitForElementVisible('button[name=btnG]', 1000)
11      .click('button[name=btnG]')
12      .pause(1000)
13      .assert.containsText('#main', 'The Night Watch')
14    }
15  };
16 };
17 
```

Write End-to-End tests in Node.js quickly and effortlessly that run against a Selenium server.

 [Download \(v0.8.16\)](#)

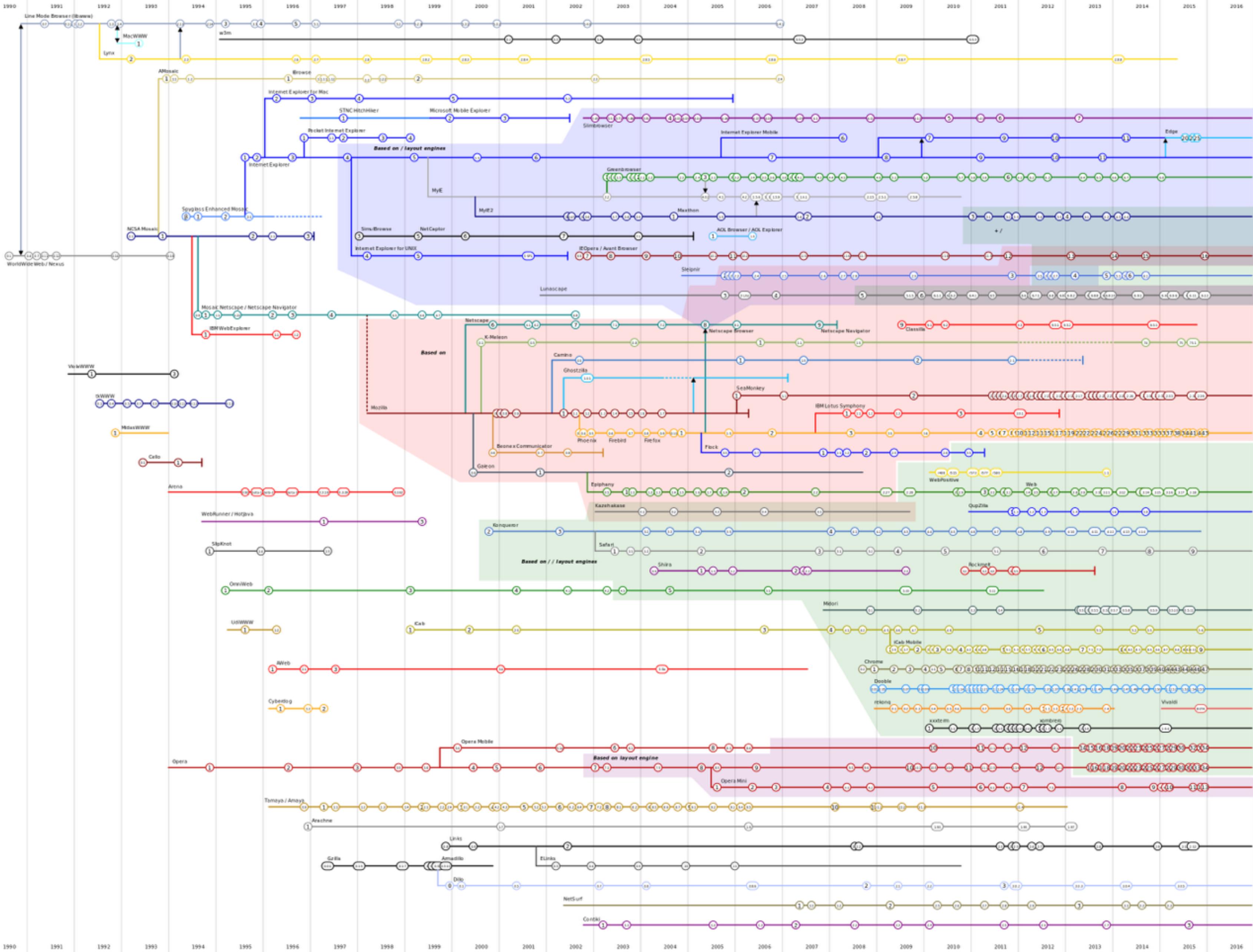
[View on GitHub](#)

[Tweet](#)  [Star](#) 3,887

# Benefits

- Essential for holistic verification
- Most realistic form of test automation
- Necessary to ensure ship-worthiness

Simply  
test all  
of these



Source: [https://en.wikipedia.org/wiki/List\\_of\\_web\\_browsers#/media/File:Timeline\\_of\\_web\\_browser\\_s.svg](https://en.wikipedia.org/wiki/List_of_web_browsers#/media/File:Timeline_of_web_browser_s.svg)



Source: <http://www.zerohedge.com/sites/default/files/images/user5464/imageroot/2014/06/All%20Hands%20On%20Deck%20-%20Clean.jpg>

# Drawbacks



fragile



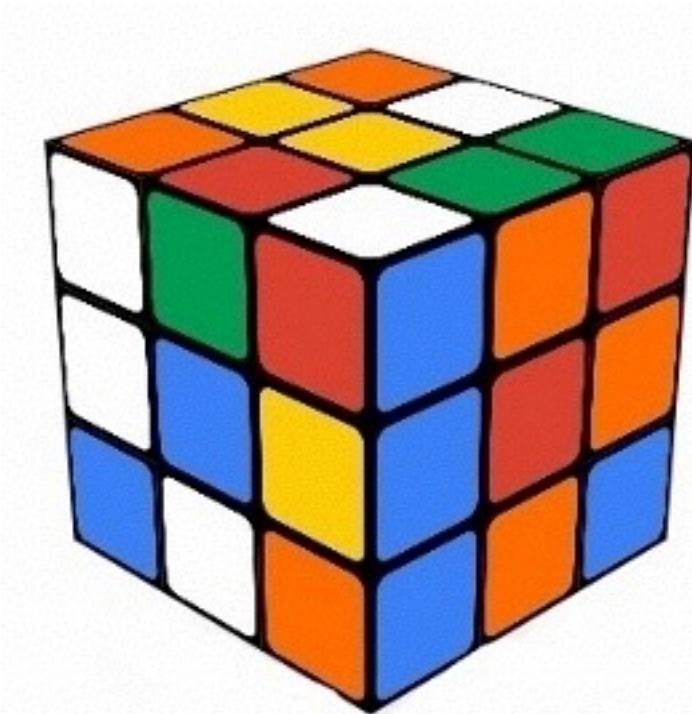
flaky



expensive

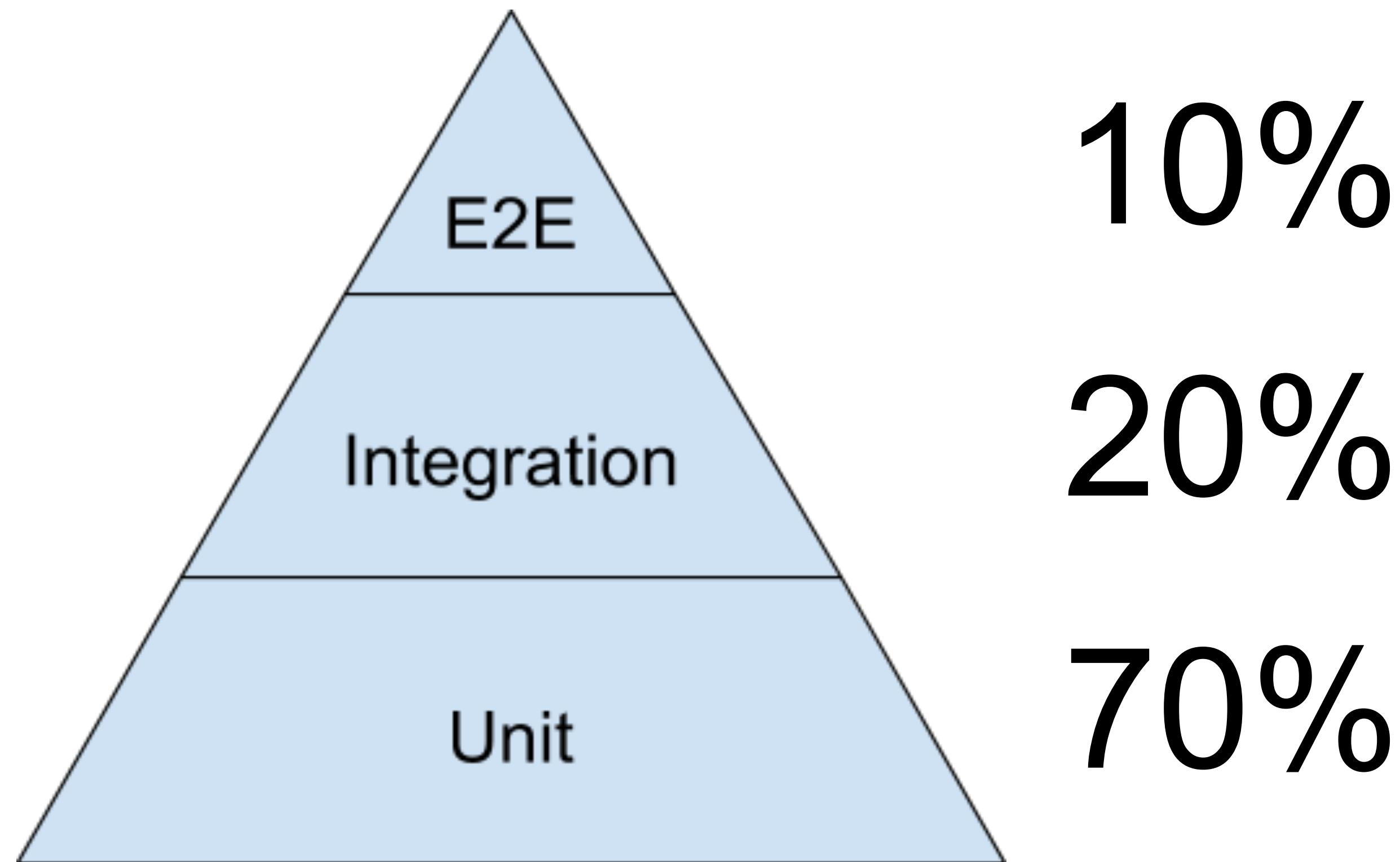


3rd party dependent



permutations

# Test Distribution



Source: Google Testing Blog <http://googletesting.blogspot.com/2015/04/just-say-no-to-more-end-to-end-tests.html>



You are  
not alone!

Source: <http://images4.fanpop.com/image/photos/1880000/MJ-You-Are-Not-Alone-you-are-not-alone-3-18855004-800-784.jpg>

#fluentconf

@JSArtisan

O'REILLY®  
Fluent

# Responsive News: BBC dev blog

## Cutting the mustard

The browser is a hostile development environment and supporting a wide range of desktop browsers can be tough work.

One of the immediate challenges we discovered when we first started the [responsive news prototype](#) was the large range of devices that we would have to support. It terrified us. This article is about a solution we use to alleviate this problem.

# #Nobody

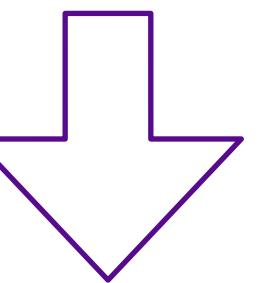
Looking at mobile devices at all - access to our service in the early stages of our project highlighted the challenge. We have ~80 significant browsers / operating system combinations regularly using our application across the globe and a long tail of hundreds more. In the four months since we first gathered the statistics there's been many more releases and updates - Amazon's Fire, Apple's new iPad, Ice Cream Sandwich, Microsoft's Mango.

To support everything on desktop we typically have to test a dozen different browsers with a multitude of different configurations and subtle differences between major and minor version numbers across three operating systems. But developers are used to this, we have coping strategies - multi-browser installs, virtual and remote environments - and it's a fairly well trodden ground.

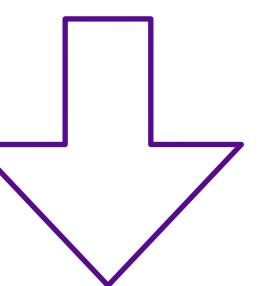
Source: <http://responsivenews.co.uk/post/18948466399/cutting-the-mustard>

# <code>

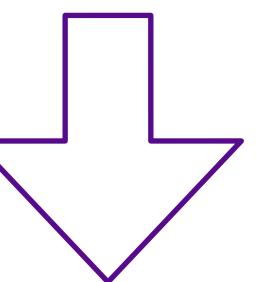
# HTML



# CSS



# JavaScript



# DOM

The screenshot shows a Mac OS X desktop with a terminal window open. The title bar of the window reads "waitForJSTest.js — /Users/brian/Documents/Fluent 2016". The terminal window displays a JavaScript file with code for end-to-end testing using the Fluent 2016 framework.

The code in the terminal window is as follows:

```
1 module.exports = {
2     'Waiting for JavaScript basic test': function(client){
3         client.url('http://www.yoursite.com/')
4         .waitForJSReady()
5         .assert.visible('.some-widget')
6         .end()
7     },
8     'Waiting for JavaScript jQuery document.ready()': function(client){
9         client.url('http://www.yoursite.com/')
10        .waitForJSReady(function(){
11            var iAmReady = false;
12            $(function(){
13                // this function runs synchronously if jQuery is ready
14                iAmReady = true;
15            });
16            return iAmReady;
17        })
18        .click('.some-button')
19        .end()
20    },
21    'Waiting for AJAX loaded test': function(client){
22        client.url('http://www.yoursite.com/')
23        .waitForJSReady(function(){
24            var contentIsLoaded = false;
25            YourAPI.friendsListLoaded.done(function(){
26                contentIsLoaded = true;
27            });
28            return contentIsLoaded;
29        })
30        .assert.containsText('.friends-list', 'Kimberly Bryant')
31        .end()
32    }
33};
```

The terminal window also shows the path "code/waitForJSTest.js" and the line number "34:1" at the bottom. The status bar at the bottom right indicates "LF UTF-8 JavaScript".



Fluent 2016

code

waitForJSTest.js

waitForJSReady.js

pageReloadTest.js

willReload.js

waitForReload.js

featureTest.js

featureTestTest.js

```
12     var isReady = false;
13     var endTime = Date.now() + timeout;
14
15     var checkJSReadiness = function() {
16         client.execute(
17             testFn || function(){
18                 // default test to determine if application is ready (JS can run)
19                 return true;
20             },
21             [],
22             function(result){
23                 isReady = (result || {}).value;
24             }
25         );
26
27         if ( ! isReady ) {
28             client.pause(1000, function(){
29                 if ( ! isReady ) {
30                     if ( Date.now() < endTime ) {
31                         // Application is not ready and we've not timed out, so run this function recursively
32                         checkJSReadiness();
33                     } else {
34                         // application is still not ready and we've timed out
35                         client.assert.ok(false, 'JavaScript was not ready after ' + timeout + ' milliseconds.');
36                     }
37                 }
38             });
39         }
40     };
41
42     // kick off the test
43     checkJSReadiness();
44 }
```

code/waitForJSReady.js 45:1

LF UTF-8 JavaScript

A screenshot of a Mac OS X desktop environment. On the left, a file browser window titled "Fluent 2016" shows a directory structure under "Fluent 2016" and "code". A file named "pageReloadTest.js" is selected. On the right, a code editor window titled "pageReloadTest.js — /Users/brian/Documents/Fluent 2016" displays the following JavaScript code:

```
1 module.exports = {
2   'Page reload test': function(client){
3     client.url('http://www.yoursite.com/upload')
4     .click('.upload-button')
5     .waitForElementVisible('#uploader')
6     // file list starts off empty
7     .willReload('.uploaded-file-list')
8     // upload a file
9     .doFileUpload('selfie.jpg')
10    // make sure upload is complete
11    .waitForReload('.uploaded-file-list')
12    // verify the list has the file in it
13    .assert.containsText('.uploaded-file-list .filename', 'selfie')
14    .end()
15  }
16};
17
```

The code implements a Mocha-style test for file upload. It navigates to a specified URL, clicks an upload button, waits for the file input field to become visible, performs a page reload, uploads a file named "selfie.jpg", and then asserts that the uploaded file's name is present in the list.

The screenshot shows a Mac OS X desktop environment with a terminal window and a code editor.

The terminal window on the left displays the following directory structure:

```
Fluent 2016
└── code
    ├── featureTest.js
    ├── featureTestTest.js
    ├── pageReloadTest.js
    ├── waitForJSReady.js
    ├── waitForJSTest.js
    └── waitForReload.js
```

The code editor on the right shows the file `willReload.js` with the following content:

```
1 exports.command = function(selector) {
2     var client = this;
3
4     var reloadClass = client.globals.reloadClass; // defined in config, e.g. "thisTooShallPass"
5
6     client.execute(
7         function(selector, reloadClass){
8             document.querySelectorAll(selector)[0].className += (' ' + reloadClass);
9         },
10        [ selector, reloadClass ]
11    );
12
13    client.waitForElementPresent(selector + '.' + reloadClass, selector + ' tagged for reload.');
14};
15
```

The status bar at the bottom of the code editor indicates the file is `code/willReload.js` at line 15:1, and the encoding is `LF UTF-8 JavaScript`.

#fluentconf

@JSArtisan

O'REILLY®  
Fluent

Fluent 2016

Fluent 2016 - End to End testing

waitForJSTest.js waitForJSReady.js pageReloadTest.js willReload.js **waitForReload.js** featureTest.js featureTestTest.js

```
1 exports.command = function(selector, callback) {  
2     var client = this;  
3  
4     callback = callback || function(){};  
5  
6     client.waitForElementNotPresent( selector + '.' + client.globals.reloadClass, callback, 'Reload complete.' );  
7 };  
8
```

/Users/brian/Documents/Fluent 2016/waitForReload.js

code/waitForReload.js 8:1 LF UTF-8 JavaScript

#fluentconf

@JSArtisan

O'REILLY®  
Fluent

featureTestTest.js — /Users/brian/Documents/Fluent 2016

Fluent 2016

code

featureTest.js

featureTestTest.js

pageReloadTest.js

waitForJSReady.js

waitForJSTest.js

waitForReload.js

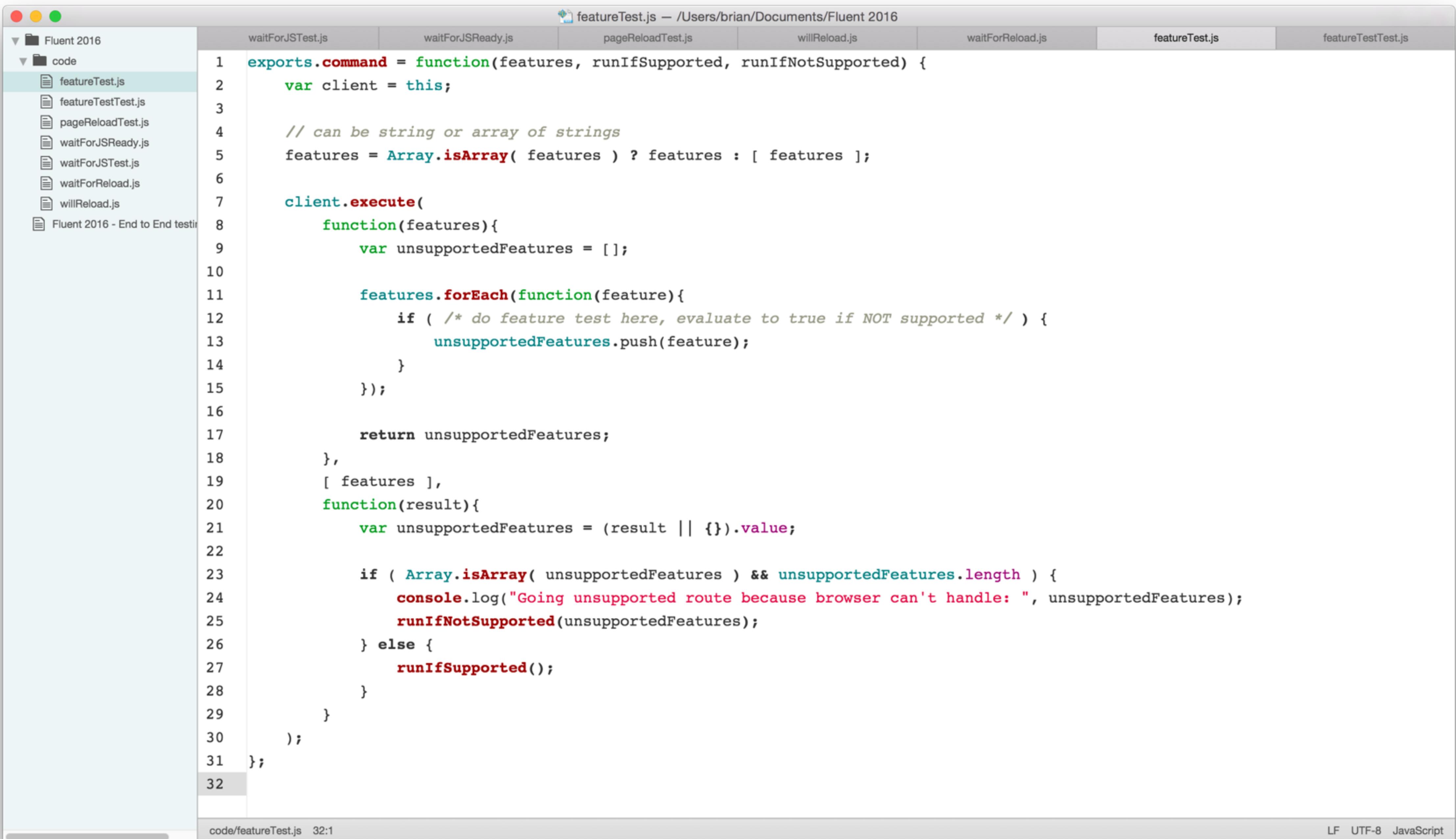
featureTest.js

featureTestTest.js

```
1 module.exports = {
2     'Feature test, basic and enhanced versions': function(client){
3         client.url('http://www.yoursite.com/uploader')
4             .featureTest(['FileReader', 'draggable'],
5                         function(){
6                             client.assert.visible('.enhanced-uploader');
7                         },
8                         function(unsupportedFeatures) {
9                             client.assert.visible('.basic-uploader');
10                        }
11                    )
12                    .end()
13                },
14    'Feature test, enhanced test only': function(client){
15        client.url('http://www.yoursite.com/imageEditor')
16            .featureTest(['FileReader', 'canvas'],
17                         function(){
18                             // test image editor here
19                         },
20                         function(unsupportedFeatures) {
21                             client.assert.ok(true, "Test skipped because " + unsupportedFeatures.join(', ') + " not supported.");
22                         }
23                     )
24                    .end()
25                }
26 };
27
```

code/featureTestTest.js 27:1

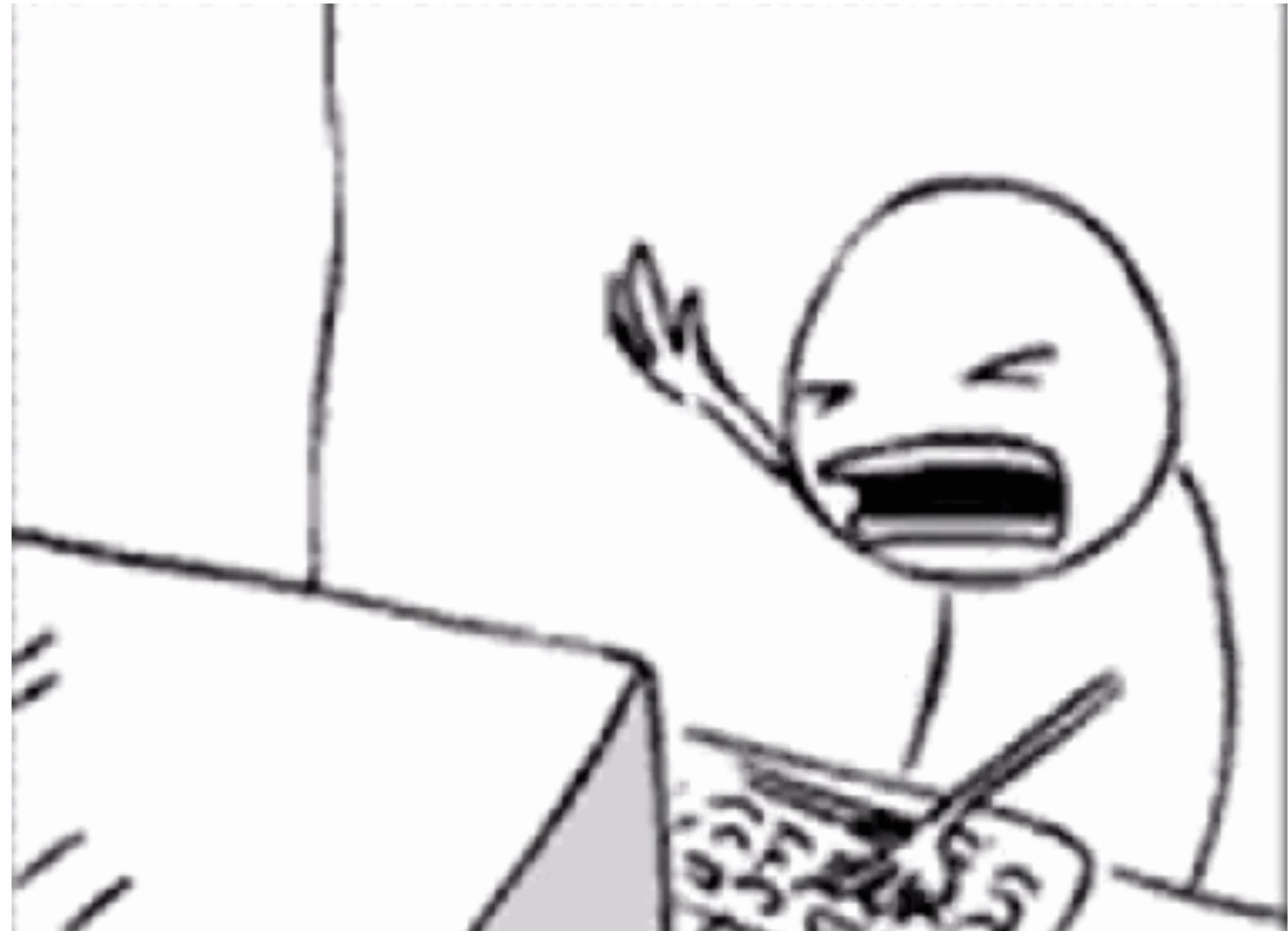
LF UTF-8 JavaScript



A screenshot of a Mac OS X terminal window titled "featureTest.js — /Users/brian/Documents/Fluent 2016". The window shows a file tree on the left and the content of "featureTest.js" on the right. The code is a JavaScript function that takes features, runIfSupported, and runIfNotSupported as arguments. It uses the client's execute method to run features and log unsupported ones. It then checks if the browser can handle the features and runs the appropriate callback.

```
1 exports.command = function(features, runIfSupported, runIfNotSupported) {
2     var client = this;
3
4     // can be string or array of strings
5     features = Array.isArray( features ) ? features : [ features ];
6
7     client.execute(
8         function(features){
9             var unsupportedFeatures = [];
10
11            features.forEach(function(feature){
12                if ( /* do feature test here, evaluate to true if NOT supported */ ) {
13                    unsupportedFeatures.push(feature);
14                }
15            });
16
17            return unsupportedFeatures;
18        },
19        [ features ],
20        function(result){
21            var unsupportedFeatures = (result || {}).value;
22
23            if ( Array.isArray( unsupportedFeatures ) && unsupportedFeatures.length ) {
24                console.log("Going unsupported route because browser can't handle: ", unsupportedFeatures);
25                runIfNotSupported(unsupportedFeatures);
26            } else {
27                runIfSupported();
28            }
29        }
30    );
31 };
32
```

</code>



Me, too  
many  
times

# A few final thoughts...

- Avoid testing for side effects
- Limit scope of end-to-end tests
- Think like a user

# craigslist is hiring!



Visit our booth for info!

or [https://www.craigslist.org/about/craigslist\\_is\\_hiring](https://www.craigslist.org/about/craigslist_is_hiring)

# Thanks!

Office hour tomorrow  
Golden Gate Room, Table C

Brian Belhumeur

Twitter: @JSArtisan