

# Advanced Testing With Jasmine

Jeff Bellsey

[jbellsy@gmail.com](mailto:jbellsy@gmail.com)

<http://futureground.net>

[https://github.com/  
jbellsy/dbc-jasmine](https://github.com/jbellsy/dbc-jasmine)



1

# SPIES



# Objectives

## Isolation

Prevent method under observation from invoking helpers (expensive, side-effect-prone)

## Tracking

Ensure those helpers are actually being called properly

## Control

Replace operations with simulations to test how our method responds

# SPIES

```
1 var MealPlan = {  
2   cost: 0,  
3  
4   // new recipe components have been chosen  
5   //  
6   updateRecipe: function updateRecipe(a,b,c) {  
7     this.cost = this.calculateCost(a,b,c);  
8   },  
9  
10  // this is an EXPENSIVE calculation! maybe does  
11  // a lot of DB querying, or AJAX requests  
12  //  
13  calculateCost: function calculateCost(a,b,c) {  
14    return a + b + c;  
15  }  
16}  
17};
```

```
// intercept ("mock") the expensive calcuation.  
// here we replace the function  
//  
spyOn(MealPlan, 'calculateCost').and.returnValue(99);
```

# SPIES

```
20 | it('should invoke "calculateCost" and store its result', function shouldInvokeRecalc() {  
21 |   // intercept ("mock") the expensive calcuation.  
22 |   // here we completely replace the function  
23 |   //  
24 |   spyOn(MealPlan, 'calculateCost').and.returnValue(99);  
25 |  
26 |   // run the update method, which should invoke the calculateCost method  
27 |   //  
28 |   MealPlan.updateRecipe(4,5,6);  
29 |  
30 |   // was our mock invoked?  
31 |   //  
32 |   expect(MealPlan.calculateCost).toHaveBeenCalledWith(4,5,6);  
33 |  
34 |   // was the value stored?  
35 |   //  
36 |   expect(MealPlan.cost).toEqual(99);  
37 | } );
```

# Useful spy methods

```
8 // creating spies
9 spyOn(obj, 'method').and.callThrough();
10 spyOn(obj, 'method').and.returnValue(0);
11 spyOn(obj, 'method').and.callFake(fn);
12 var newSpy = jasmine.createSpy('mySpy');

13
14 // testing spies
15 expect(obj.method.calls.count()).toEqual(1);
16 expect(obj.method).toHaveBeenCalledWith('param1', 'param2');

17
18 var args = obj.method.calls.mostRecent().args;
19 expect(args[0]).toBeLessThan(7);
```

2

# TIMERS



# TIMERS

```
1  'use strict';  
2  
3  var myAnimation = {  
4  
5      animationLength: 1000,  
6      hasAnimationRun: false,  
7  
8      run: function RunAnimation() {  
9          var self = this;  
10  
11          // this pretend animation is super-simple:  
12          // (1) pause, then (2) flag the animation as completed  
13          //  
14          setTimeout(function AnimationCallback() {  
15              self.hasAnimationRun = true;  
16          }, self.animationLength)  
17      }  
18  };
```

```
jasmine.clock().install();  
jasmine.clock().tick(ms);  
jasmine.clock().uninstall();
```

# TIMERS

```
6  describe('Animation', function AnimationTest() {
7
8      // install mock timer
9      //
10     beforeEach(function BeforeAnimationTests() {
11         jasmine.clock().install();
12     });
13     afterEach(function AfterAnimationTests() {
14         jasmine.clock().uninstall();
15     });
16
17     it('should not run the animation synchronously', function shouldNotRunAnimSync() {
18
19         myAnimation.run();
20         expect(myAnimation.hasAnimationRun).toBeFalsy();
21     });
22
23     it('should run the animation Asynchronously', function shouldRunAnimAsync() {
24
25         // initiate an async event
26         //
27         myAnimation.run();
28
29         // move the mock timer forward the right amount
30         //
31         jasmine.clock().tick(myAnimation.animationLength);
32
33         // and NOW it should have completed
34         //
35         expect(myAnimation.hasAnimationRun).toBeTruthy();
36     });
37 });
```

# TIMERS

What about  
requestAnimationFrame?

Velocity.js

`$.Velocity.mock = true;`

Greensock

`TimelineLite.timeScale()`

Some testing support is best  
provided outside of Jasmine

2

Bonus!

# AJAX



# AJAX

Most of the time...  
YOU DON'T.

## Use library mocks

Angular.js => angular-mocks  
jQuery => mockAjax  
Backbone => backbone-faux-server

## Integration tests

Karma  
Selenium  
PhantomJS

But once in a while...

# AJAX

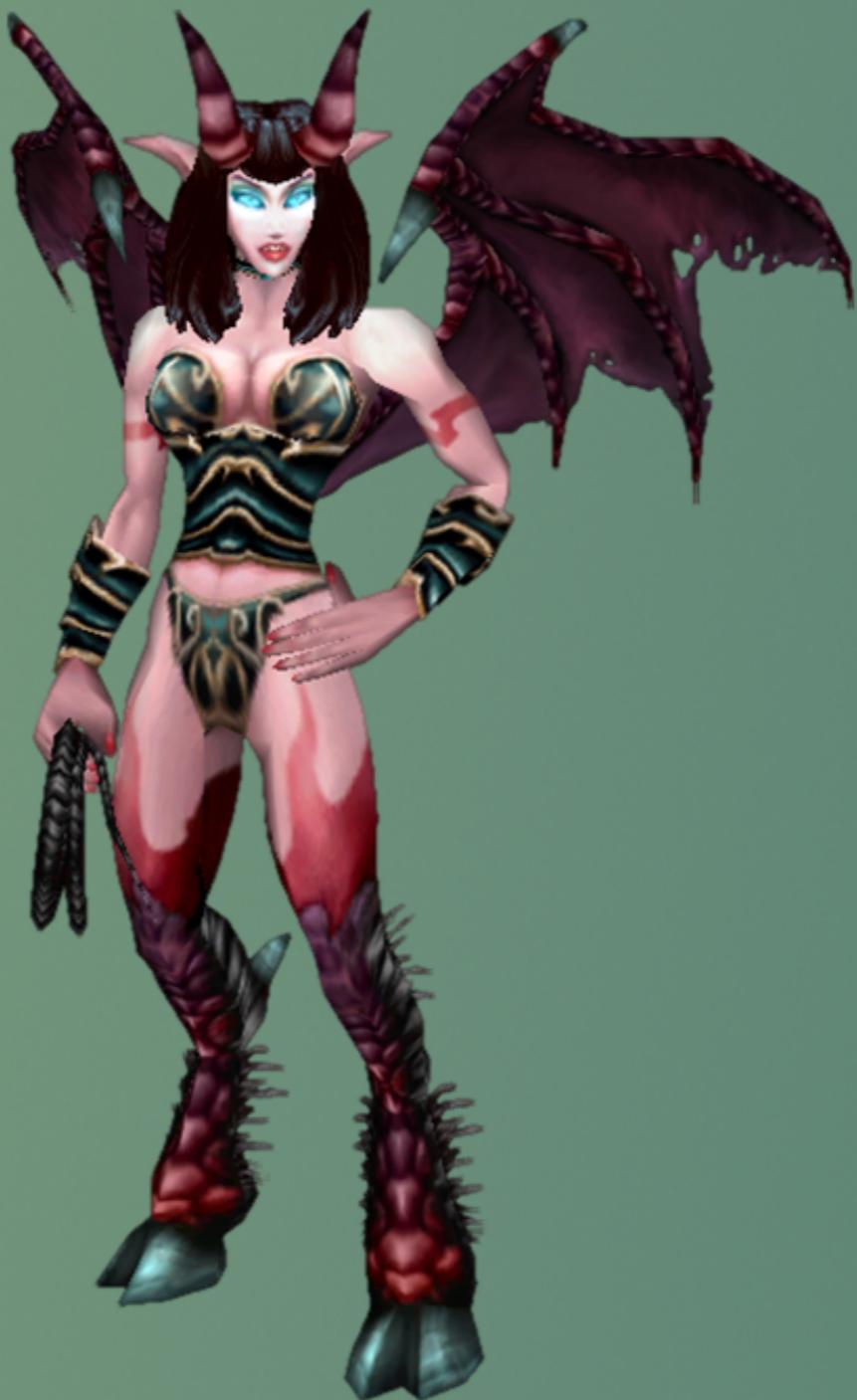
```
1  'use strict'; /* global $ */
2
3  var OffsiteAPI = {
4
5      getData: false,
6
7      // offsite asynchronous API call to randomuser.me
8      //
9      retrieveData: function RetrieveData() {
10          var self = this;
11
12          // use the promise interface to $.ajax.
13          // the return value is also a promise
14          //
15          return $.ajax('http://api.randomuser.me/')
16              .done(function AjaxDoneCallback(data) {
17                  self.getData = true;
18              });
19      }
20  };
```

# AJAX

```
4  'use strict'; /* global $, describe,it,expect */
5
6  describe('Ajax', function AjaxTest() {
7
8      // note the use of the "onComplete" callback we're given
9      //
10     it("successfully tests asynch API call", function AsyncTests(onComplete){
11
12         // initiate an AJAX request, which passes
13         // back a jQuery promise
14         //
15         var $promise = OffsiteAPI.retrieveData();
16
17         $promise
18             .done(function PromiseDone(data) {
19
20                 // validate the data here:
21                 //
22                 expect(data).toBeDefined();
23             })
24             .always(onComplete);    // <= this is the key; it closes the test
25         });
26     });
});
```

3

# DOM



# DOM

```
3 var DOMactions = {  
4  
5     // this method sticks an image into a parent container.  
6     //  
7     insertKittenInto: function(parent) {  
8         var img = '';  
9         $(parent).append(img);  
10    }  
11};
```

# Jasmine-jQuery plugin

```
6 describe('DOM Actions', function DOMActionTestSuite() {  
7  
8     // set up the DOM required by our "plugin".  
9     // no need to clean it up between tests.  
10    //  
11    beforeEach(function BeforeDOM() {  
12        setFixtures('<div id="imageContainer"></div>');  
13    });  
14  
15    it('should insert an image tag with class "kitten"', function InsertImageTest() {  
16        DOMactions.insertKittenInto('#imageContainer');  
17        expect($('#imageContainer')).toContainElement('img.kitten');  
18    });  
19});
```

# DOM

## Jasmine-jQuery plugin

```
25  expect($div).toExist();
26  expect($div).toBeEmpty();
27  expect($div).toBeVisible();
28  expect($div).toHaveClass('active');
29  expect($h1).toContainText('title');
30  expect($checkbox).not.toBeChecked();

31
32 // event spies
33 eventSpy = spyOnEvent('#buy-button', 'click');
34 $btn.trigger('click');
35 expect('click').toHaveBeenTriggeredOn('#buy-button');
```

Has tons of great matchers

# 4

# CUSTOM MATCHERS



# CUSTOM MATCHERS

```
1  'use strict';
2
3  var BookShelf = {
4
5      books: [
6          'Alphabet Soup',
7          'Infinite Jest',
8          'Wisdom of the Enneagram'
9      ],
10
11      addBook: function(b) {
12          this.books.push(b);
13          this.books.sort();
14      }
15  };
```

# CUSTOM MATCHERS

```
6   // simple, generic test
7   function isArraySorted(arr) {
8     for (var i = 1, ct = arr.length; i < ct; ++i) {
9       if (arr[i-1] > arr[i])
10         return false;
11     }
12   }
13 }
14
15 describe('Custom Matcher', function CustomMatcherSuite() {
16
17   var customMatchers = {
18
19     toBeSorted: function ToBeSorted() {
20       return {
21         compare: function(actualArray) {
22           var isArraySorted = isArraySorted(actualArray);
23           return {
24             pass: isArraySorted,
25             message: isArraySorted
26               ? "Expected [" + actualArray + "] to be sorted"
27               : "Expected [" + actualArray + "] to be sorted, but it wasn't"
28           };
29         };
30       };
31     };
32   };
33
34   beforeEach(function BeforeCustom() {
35     jasmine.addMatchers(customMatchers);
36   });
37
38   it('should insert a book and stay sorted', function InsertBookTest() {
39     BookShelf.addBook("Silmarillion");
40     expect(BookShelf.books).toBeSorted();
41   });
42});
```



Jeff Bellsey

[jbellsey@gmail.com](mailto:jbellsey@gmail.com)

<http://futureground.net>

<https://github.com/jbellsey/dbc-jasmine>