UNIT TESTING

Why Test ?

Scenario(Form Validation)

```
describe('user login form', function() {
    // critical
    it('ensure invalid email addresses are caught', function() {});
    it('ensure valid email addresses pass validation', function() {});
    it('ensure submitting form changes path', function() { });

    // nice-to-haves
    it('ensure client-side helper shown for empty fields', function() { });
    it('ensure hitting enter on password field submits form', function() { });
});
```

What is testable code?

Separation of concerns

Object-oriented code design

Loose coupling / dependency injection

Elimination of globals

Clear code and documentation

Jasmine Basics

Refer :

- jasmineStructure.js

Basic Example

- Calculator
  - Refer Calculator

Angular Unit Testing

Angular is built keeping testing in mind

Spies

Jasmine has test double functions called spies.

You can spy functions and then you will be able to assert a couple of things about it - you can check if it was called, what parameters it had, if it returned something or even how many times it was called!

Most vital part in unit testing.

Spies

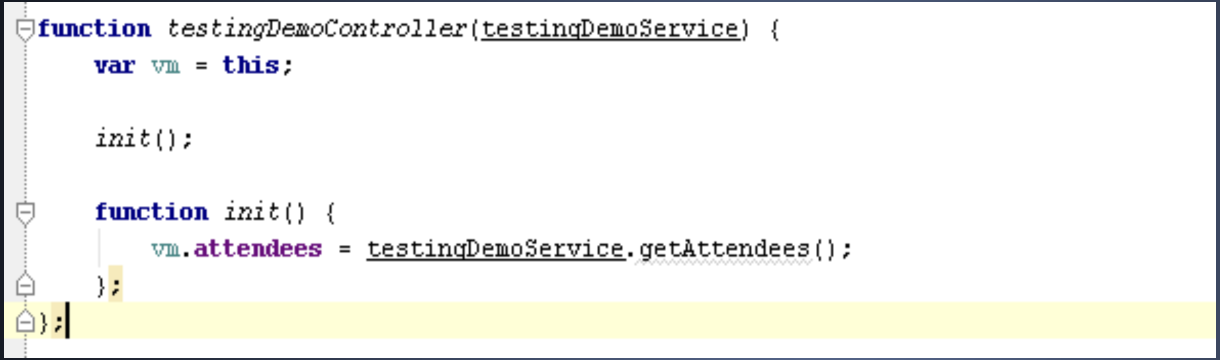
spyOn

and.returnvalue (andReturn)

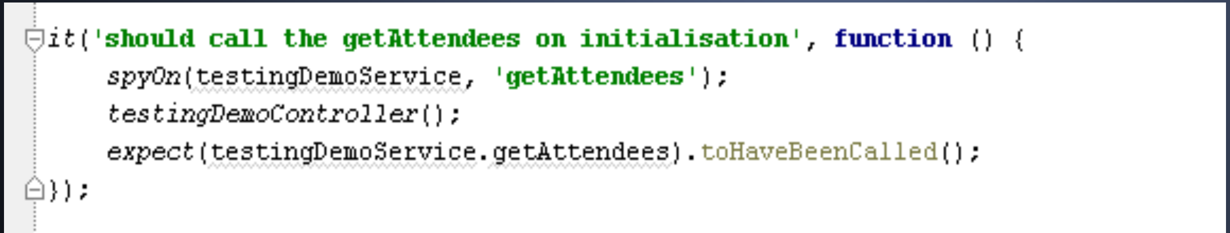
and.callThrough (andCallThrough)

createSpy

spyOn

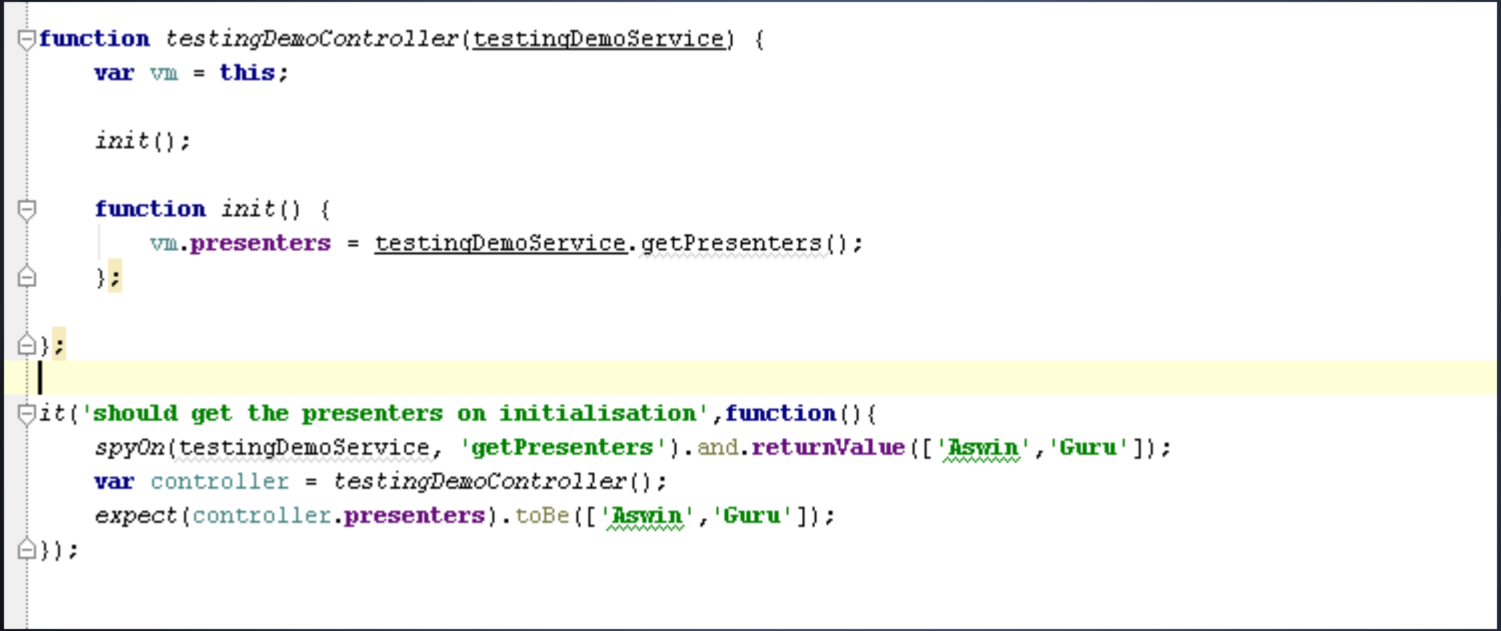


```
function testingDemoController(testingDemoService) {
    var vm = this;

    init();

    function init() {
        vm.attendees = testingDemoService.getAttendees();
    };
};
```



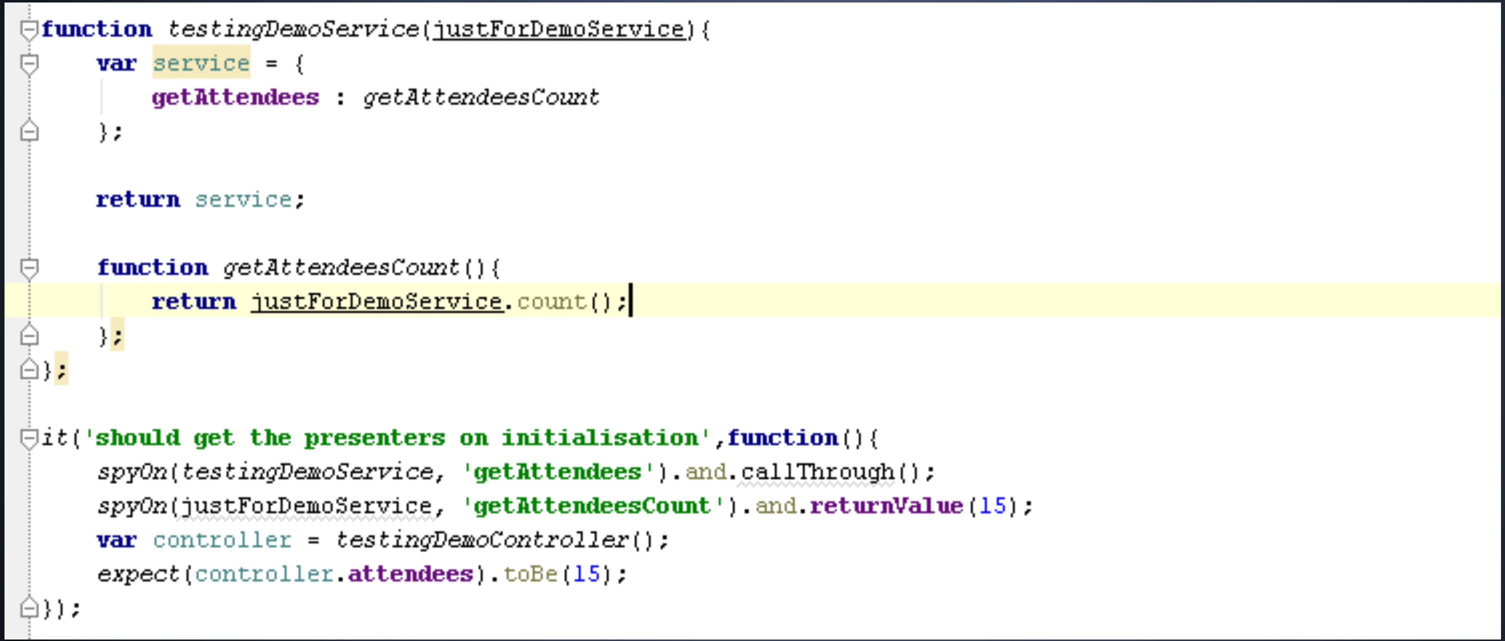
```
it('should call the getAttendees on initialisation', function () {
    spyOn(testingDemoService, 'getAttendees');
    testingDemoController();
    expect(testingDemoService.getAttendees).toHaveBeenCalled();
});
```

and.returnValue



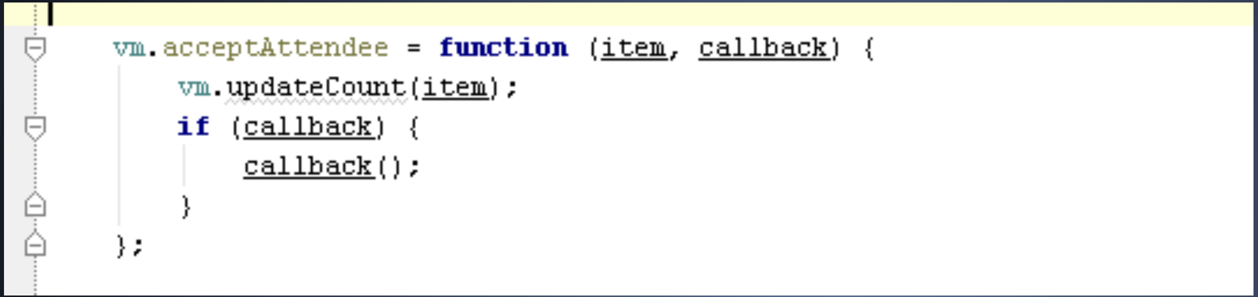
```
function testingDemoController(testingDemoService) {
    var vm = this;

    init();

    function init() {
        vm.presenters = testingDemoService.getPresenters();
    };

};

it('should get the presenters on initialisation',function(){
    spyOn(testingDemoService, 'getPresenters').and.returnValue(['Aswin','Guru']);
    var controller = testingDemoController();
    expect(controller.presenters).toBe(['Aswin','Guru']);
});
```

and.callThrough

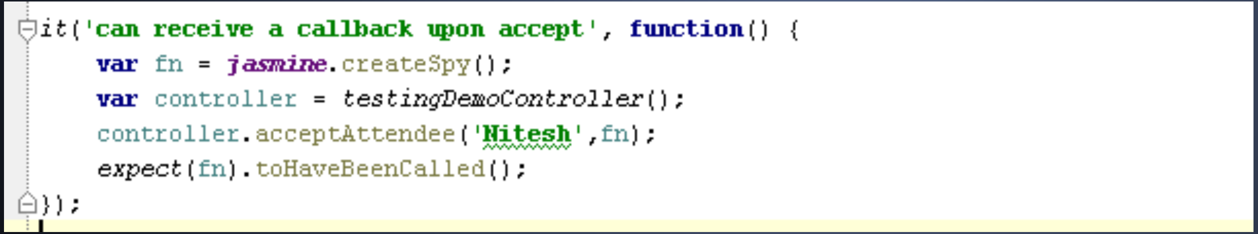


```
function testingDemoService(justForDemoService){
    var service = {
        getAttendees : getAttendeesCount
    };

    return service;

    function getAttendeesCount(){
        return justForDemoService.count();
    };
};

it('should get the presenters on initialisation',function(){
    spyOn(testingDemoService, 'getAttendees').and.callThrough();
    spyOn(justForDemoService, 'getAttendeesCount').and.returnValue(15);
    var controller = testingDemoController();
    expect(controller.attendees).toBe(15);
});
```

createSpy



```
vm.acceptAttendee = function (item, callback) {
    vm.updateCount(item);
    if (callback) {
        callback();
    }
};
```



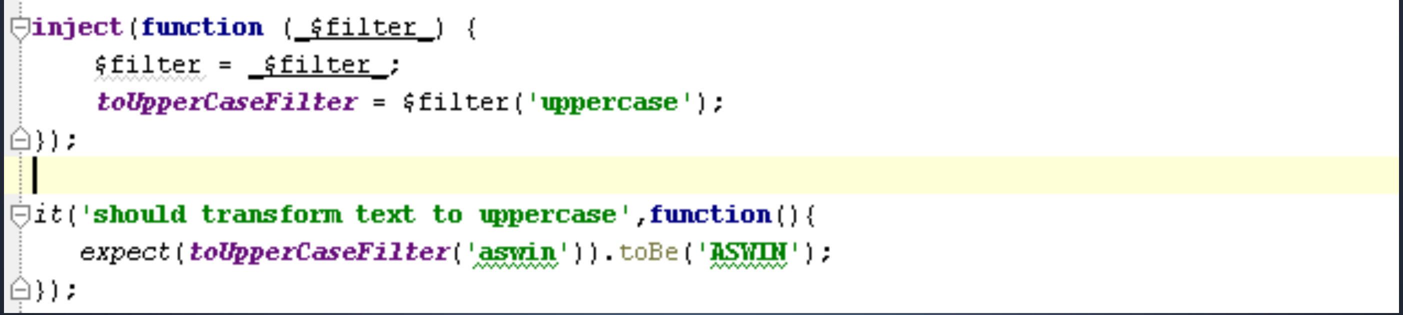
```
it('can receive a callback upon accept', function() {
    var fn = jasmine.createSpy();
    var controller = testingDemoController();
    controller.acceptAttendee('Nitesh',fn);
    expect(fn).toHaveBeenCalled();
});
```

Filters

Easiest component to test.

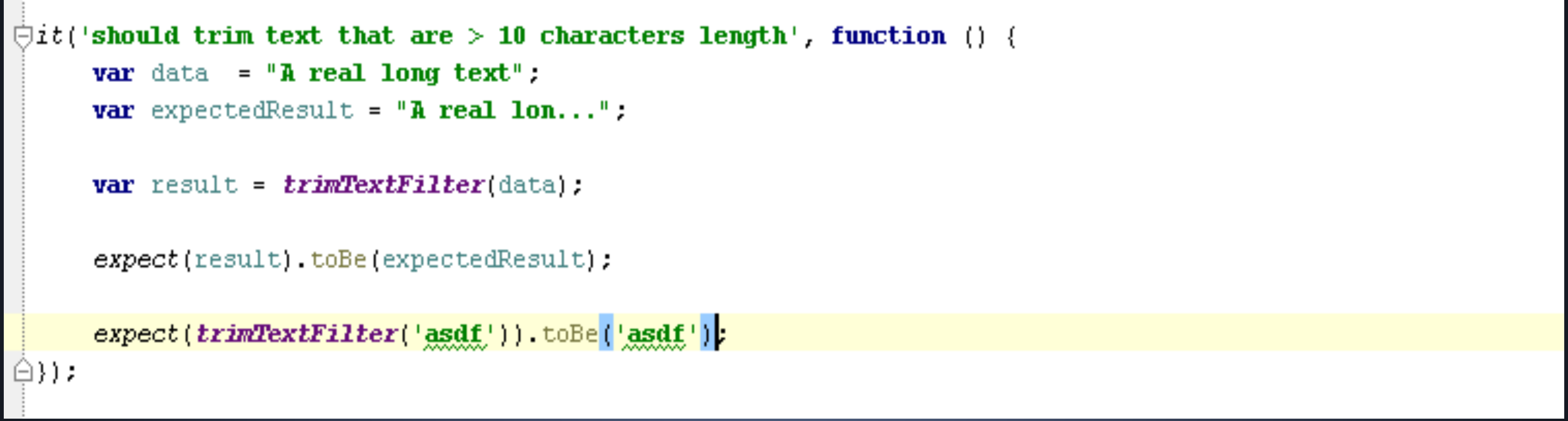
Just invoke the filter and expect the result.

Filters



```
inject(function (_$filter_) {
    $filter = _$filter_;
    toUpperCaseFilter = $filter('uppercase');
});

it('should transform text to uppercase',function(){
   expect(toUpperCaseFilter('aswin')).toBe('ASWIN');
});
```

Filters



```
it('should trim text that are > 10 characters length', function () {
    var data  = "A real long text";
    var expectedResult = "A real lon...";

    var result = trimTextFilter(data);

    expect(result).toBe(expectedResult);

    expect(trimTextFilter('asdf')).toBe('asdf');
});
```

Services

Refer to:

- account.services.js

Controllers

Refer to:

- dashboard.controller.spec.js

Next Steps

- Angular directives
- Custom matchers

Key take away

SEPARATION OF CONCERNS

(can’t stress enough on the importance of this)

STEP-BY-STEP APPROACH

BASICS OF ANGULAR TESTING

References

- https://docs.angularjs.org/guide/unit-testing
- http://jasmine.github.io/2.0/introduction.html
- http://angular-tips.com/blog/categories/unit-test/
- http://www.pluralsight.com/training/Player?author=jesse-liberty&name=tafs-m4&clip=4&course=testing-angularjs-from-scratch
- http://www.pluralsight.com/training/Player?author=misko-hevery&name=code-testability-m1&clip=5&course=code-testability

THANK YOU