Software Testing: Unit Testing and Acceptance Testing

Bulletproofing your application

Bulletproofing your application

William Chan
Lead Platform Engineer, 605.tv, Capital One, FreeWheel (Comcast)

Objectives

- Learn what the different types of testing are and what they test
- Learn what tests are helpful when time is a constraint
- Learn how to unit test properly with mocks and stubs

What is software testing?

Testing software helps build in quality into the application and helps prevent bugs from leaking into software as new features are developed.

3

What are the benefits of testing your software?

- It helps you pinpoint and debug the component and the precise location of where an error may have occurred
- It helps with **regression**; to ensure that delivery of new features are intended to have a conflict with existing features

What is exploratory testing?

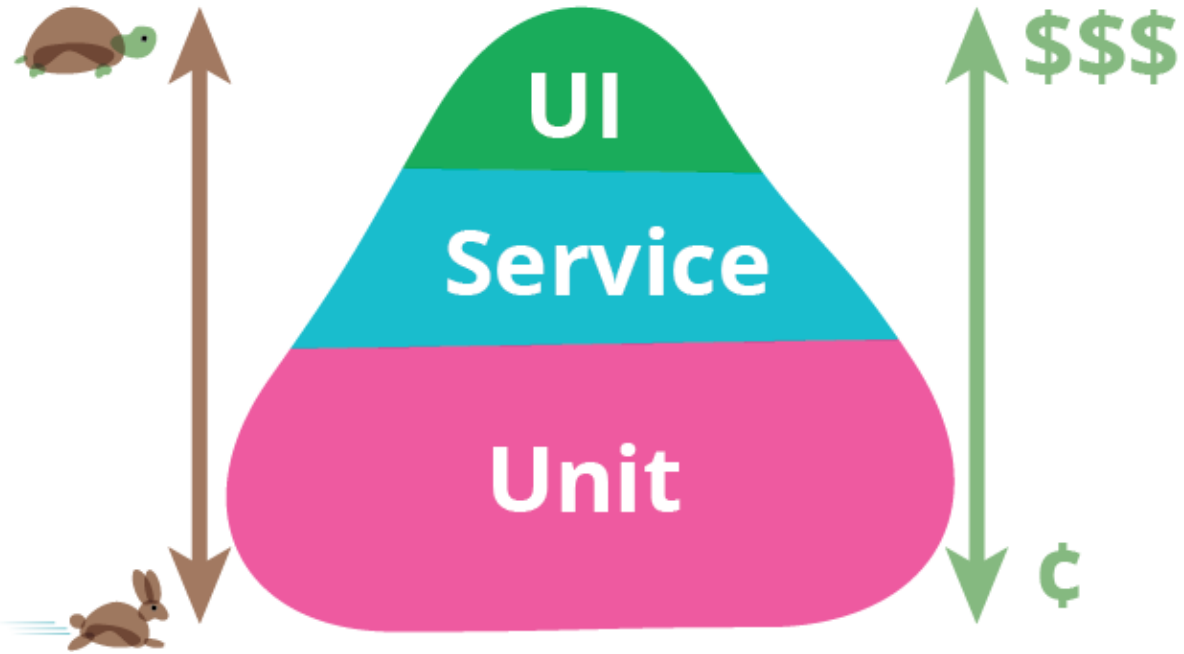
Exploratory testing is a stimulating activity that documents how a user uses the product and exploring use cases that help automate tests. Often this helps with discovering non-obvious errors.

5

Different Types of Testing

- Unit tests - low level, close to the source of the application
- Integration tests - tests the integration of modules and services in the application that work in conjunction with one another
- Functional tests - focuses on business requirements and tests the functions of the application
- End-to-end tests - tests the software in a complete application environment
- Acceptance tests - tests that the software satisfies business requirements
- Performance tests - tests that the performance characteristics under significant load
- Smoke tests - tests that basic functionality of the application; are usually quick to run in nature

Testing Pyramid



Tests are more costly in terms of run time and complexity as it integrates with more components

Source: https://martinfowler.com/articles/practical-test-pyramid.html

7

Unit Testing

8

What is Unit Testing?

Unit testing are testing **individual** units of code.

A unit of code is an object or a behavior in the code. In a unit test, dependencies are typically **mocked** or **stubbed** to isolate the test to test the unit instead of testing it in integration with its dependencies.

9

What are Stubs?

Stubs simulate the behavior of an object. It allows for easily testing different **cases** by testing forcing code to step into different branches of control flow structures.

10

What are Mocks?

Mocks are similar to stubs but with the difference that they expect that it has been used correctly.

11

Unit Testing Principles

- Rely on interfaces instead of concrete types so that you can mock out dependencies
- Rely on dependency injection instead of creating the type within the function's methods

12

Acceptance Testing

13

What is Acceptance Testing and End-to-End testing?

What is Acceptance Testing and End-to-End testing?

Acceptance testing are a general form of end-to-end tests. Acceptance tests are generally written as a specification that the product owner reads and can use to accept and mark the stories as complete.

Depending on the size of the system they can usually be larger.

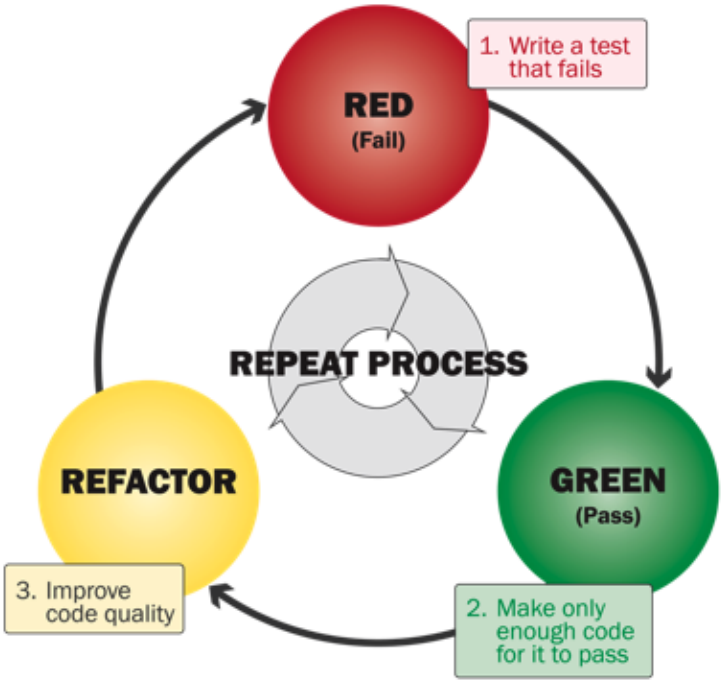
Test Driven Development (TDD)

1. Write tests as a specification for intended application components
2. Write code in the components to pass the tests that were written

Red/Green Testing

1. Write the simplest test case for iterating on the feature that fails
2. Write the simplest code that passes the test cases

Red/Green Testing Cont'd



Source: http://ryantablada.com/post/red-green-refactor---a-tdd-fairytale

Behavior Driven Development (BDD)

Testing behaviors of the tests

An example in Jasmine (JavaScript)

```
describe("MathUtils", function() {
    var calc;

    //This will be called before running each spec
    beforeEach(function() {
        calc = new MathUtils();
    });

    describe("when calc is used to peform basic math operations", function(){

        //Spec for sum operation
        it("should be able to calculate sum of 3 and 5", function() {
            expect(calc.sum(3,5)).toEqual(8);
        });
    });
});
```

17

Testing Tools

18

Unit Testing Tools

JavaScript

- Jasmine, Mocha, Chai, Sinon

Java

- JUnit, Mockito, Spock

Python

- unittest, pytest, nose

Ruby

- RSpec, Test::Unit

19

Acceptance Testing Tools

- Cucumber - uses a syntax called **Gherkin** that reads similar to English
- Selenium - web browser automation testing framework to test user behavior
- PhantomJS - headless browser testing to test front end features and behaviors

20

Sample Gherkin Syntax

Source: http://docs.behat.org/en/v2.5/guides/1.gherkin.html

```
Feature: Serve coffee
In order to earn money
Customers should be able to
buy coffee at all times
```

```
Scenario: Buy last coffee
    Given there are 1 coffees left in the machine
    And I have deposited 1 dollar
    When I press the coffee button
    Then I should be served a coffee
```

21

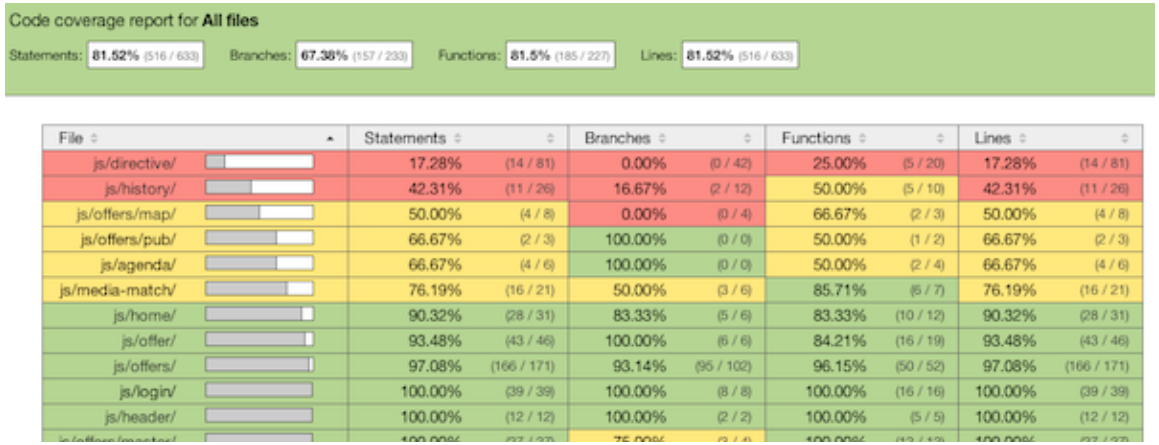
Testing Metrics

Coverage

- Lines - percentage of lines that are covered
- Statement - percentage of statements covered; statements can sometimes span multiple lines
- Branches - percentage of branches that are covered

- Functions - percentage of functions that are covered

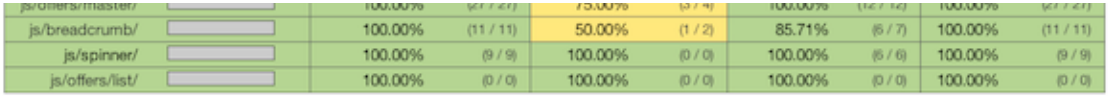
Sample Test Coverage Report



Code coverage report for **All files**

Statements: **81.52%** (516 / 633) Branches: **67.38%** (157 / 233) Functions: **81.5%** (185 / 227) Lines: **81.52%** (516 / 633)

File	Statements		Branches		Functions		Lines	
js/directive/	17.28%	(14 / 81)	0.00%	(0 / 42)	25.00%	(5 / 20)	17.28%	(14 / 81)
js/history/	42.31%	(11 / 26)	16.67%	(2 / 12)	50.00%	(5 / 10)	42.31%	(11 / 26)
js/offers/map/	50.00%	(4 / 8)	0.00%	(0 / 4)	66.67%	(2 / 3)	50.00%	(4 / 8)
js/offers/pub/	66.67%	(2 / 3)	100.00%	(0 / 0)	50.00%	(1 / 2)	66.67%	(2 / 3)
js/agenda/	66.67%	(4 / 6)	100.00%	(0 / 0)	50.00%	(2 / 4)	66.67%	(4 / 6)
js/media-match/	76.19%	(16 / 21)	50.00%	(3 / 6)	85.71%	(6 / 7)	76.19%	(16 / 21)
js/home/	90.32%	(28 / 31)	83.33%	(5 / 6)	83.33%	(10 / 12)	90.32%	(28 / 31)
js/offer/	93.48%	(43 / 46)	100.00%	(6 / 6)	84.21%	(16 / 19)	93.48%	(43 / 46)
js/offers/	97.08%	(166 / 171)	93.14%	(95 / 102)	96.15%	(50 / 52)	97.08%	(166 / 171)
js/login/	100.00%	(39 / 39)	100.00%	(8 / 8)	100.00%	(16 / 16)	100.00%	(39 / 39)
js/header/	100.00%	(12 / 12)	100.00%	(2 / 2)	100.00%	(5 / 5)	100.00%	(12 / 12)



js/breadcrumb/	100.00%	(11 / 11)	50.00%	(1 / 2)	85.71%	(6 / 7)	100.00%	(11 / 11)
js/spinner/	100.00%	(9 / 9)	100.00%	(0 / 0)	100.00%	(6 / 6)	100.00%	(9 / 9)
js/offers/list/	100.00%	(0 / 0)	100.00%	(0 / 0)	100.00%	(0 / 0)	100.00%	(0 / 0)

Source: http://revolunet.github.io/blog/2013/12/05/unit-testing-angularjs-directive/

23

Summary

- Unit testing techniques using mocks and stubs to isolate testing a unit of code rather than code in integration
- Acceptance testing can help verify product and business requirements
- The more integrated a test is, the more complex and time required to run the test
- Test coverage metrics to get an overview of where tests may be lacking

24

Thank you

William Chan
Lead Platform Engineer, 605.tv, Capital One, FreeWheel (Comcast)
http://linkedin.com/in/wchan2 (http://linkedin.com/in/wchan2)