# MOCKING AND STUBBING IN RSPEC


March 8, 2016

By Emily Fan / @emilyjustinefan

# UNIT TESTING

A software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation.

# ISOLATED TESTING

*“In order to achieve isolated testing, you necessarily have to fake part of the world that [your] test is going to touch.”*

*- Sam Phippen, 2014*

# RSPEC::MOCKS

Test-double framework for RSpec with support for method stubs, fakes, and message expectations on generated test-doubles and real objects alike.

```
gem install rspec       # rspec-core, rspec-expectations, rspec-mocks
gem install rspec-mocks # rspec-mocks only
```

# TEST DOUBLES

An object that stands in for another object in your system during a code example.

```
stand_in_object = double('stand-in-object')
```

# METHOD STUBS

A method stub is an implementation that returns a pre-determined value

```
class ReportCard
  def initialize(student)
    @student = student
  end

  def average
    grades = @student.grades
    grades.inject{ |sum, el| sum + el }.to_f / grades.size
  end
end
```

# METHOD STUBS

Inside report_card_spec.rb

```
describe '#average' do
  it 'calculates the student\'s average grade' do
    student = double('student')
    allow(student).to receive(:grades).and_return([80, 90, 70, 80])
    report_card = ReportCard.new(student)

    expect(report_card.average).to eq(80)
  end
end
```

# MESSAGE (MOCK) EXPECTACTIONS

An expectation that an object should receive a message

```
class ReportCard
  def generate(student)
    Logger.log_report_card(student) if average <= 50
  end
  .
  .
  .
end
```

```
class Logger
  def self.log_report_card(student)
    # some code that logs stuff
  end
end
```

# MESSAGE (MOCK) EXPECTACTIONS

Inside report_card_spec.rb

```
describe '#generate' do
  context 'when the student is failing' do
    it 'logs a message' do
      student = double('student', :grades => [30, 40, 50, 40])
      expect(Logger).to receive(:log_report_card).with(student)
      report_card = ReportCard.new(student)

      report_card.generate(student)
    end
  end

  context 'when the student is passing' do
    it 'does not log a message' do
      student = double('student', :grades => [80, 70, 80, 90])
      expect(Logger).to_not receive(:log_report_card)
      report_card = ReportCard.new(student)

      report_card.generate(student)
```

# NOMENCLATURE

## MOCKS ≠ STUBS

**Test Stub** = a Test Double that *only* supports method stubs

**Mock Object** = a Test Double that supports message expectations *and* method stubs

# TEST PATTERNS: CHANGES IN FLOW

Typical test pattern: **arrange-act-asert**

Mock/Stubbing test pattern: **arrange-assert-act**

**Test spies** maintain the typical pattern

# USAGE GUIDELINES

1. Isolation from dependencies
2. Isolation from nondeterminism
3. Making progress without implementing dependencies
4. Interface discovery
5. Focus on roles and interactions

# RISKS AND TRADEOFFS

1. Over specialization
2. Nested doubles
3. Absence of coverage
4. Brittle examples

# RESOURCES

- RSpec Mocks - Github README
- RSpec Mocks 3.4 - Documentation
- An Introduction to Spies in RSpec - Sam Phippen (2014)
- Stubs, Mocks and Spies in RSpec - Simon Coffey (2016)
- Why You Don't Get Mock Objects - Gregory Moeck (2011)
- The RSpec Book - David Chelimsky (2010)
- Mock Roles Not Objects - Steve Freeman, Nat Pryce, Tim Mackinnon, Joe Walnes (2004)
- Mocks Aren't Stubs - Martin Fowler (2007)

https://github.com/emilyjfan/rspec-mocks-presentation

@emilyjustinefan