YYC Ruby Meetup

March 2013

Since last time...

- New Ruby version! ZOMG!!!
  - Ruby-2.0.0-p0 into the wild
- Rails Security Fix, 3.2.13, March 18 2013
- A more interesting Rails Version... well Beta
  - Rails 4.0.0-beta1

- At **LEAST** 30 days since the last Raptor attack

"Science"

700+ test Rails Rspec suite

Ruby-2.0.0-p0 - 23.822 / avg.
Ruby-1.9.3-p374 - 26.576 / avg.

Testing in Ruby

Testing...

Do it!

Fin

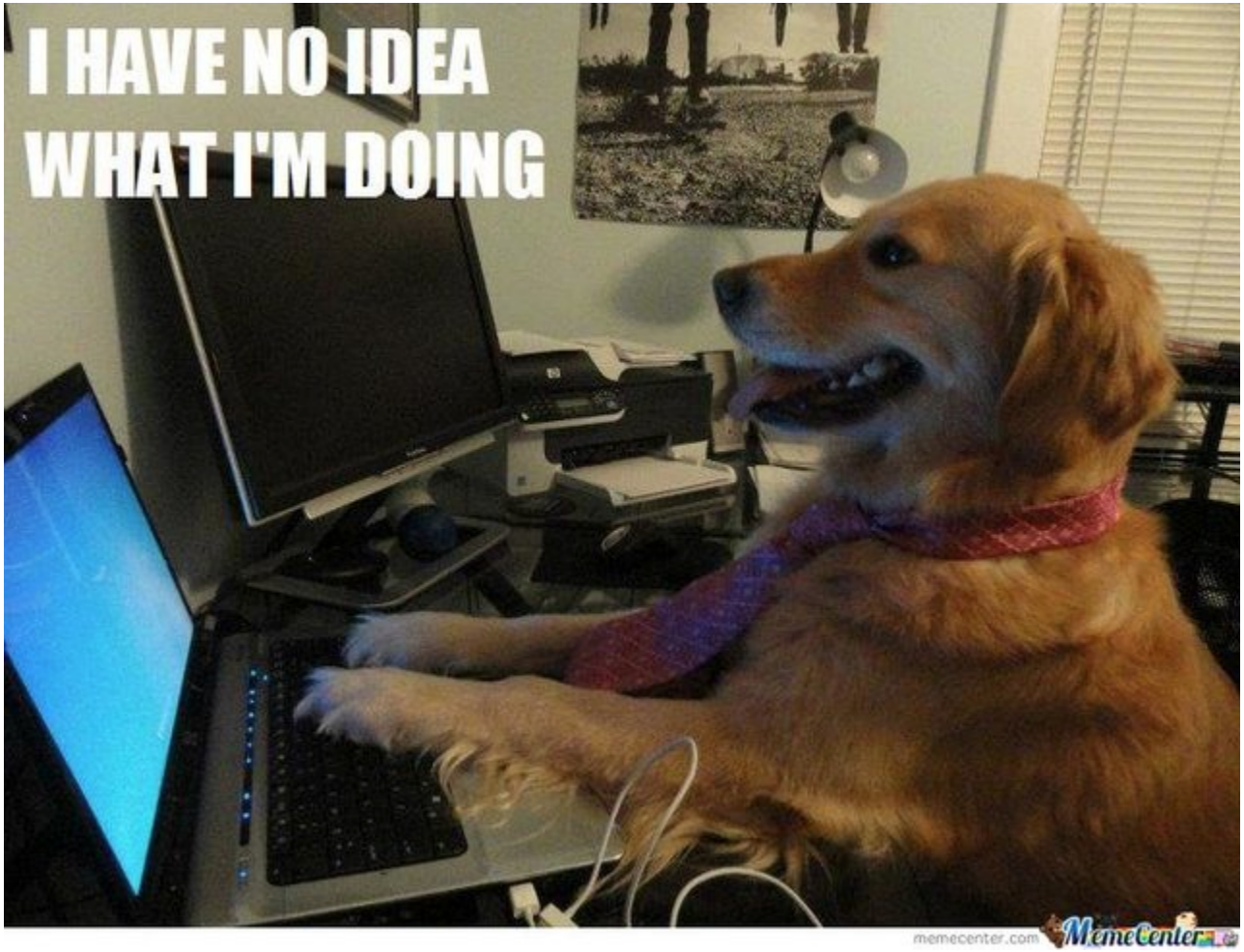
Testing in Ruby: A Rope of Sand

Ben Stevenson
Geek
Web Developer
Dev @ Wrangle HR

begin

- Calgarian
- U of C Grad
  - CPSC / SENG
- Java(Spring, WebObjects), .NET(C#)
  - Ruby for the past 6, full-time+ the past 2
- Things I love
  - Code
  - Zombies
  - Craft Beer, IPA's

My initials are BS...



Hard time writing this talk

- A wealth of information
- TDD is in wide acceptance
- Personal

So What?

- Testing through the years
- Big Deals
- Relating Frameworks/Practices to your lives
- A few words on my own methodology
  - And, why it doesn't matter

Why do we test?

We were told we have to...

I feel very bad for you...

Have we mentioned it's awesome?

No really!

/salespitch

Because it enforces correctness...

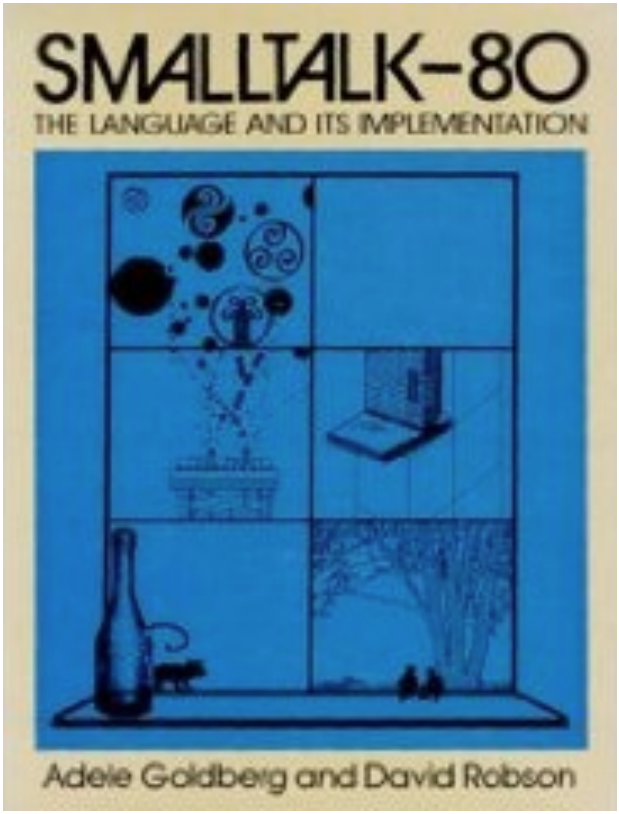
Yes, yes it does!

Trivia!

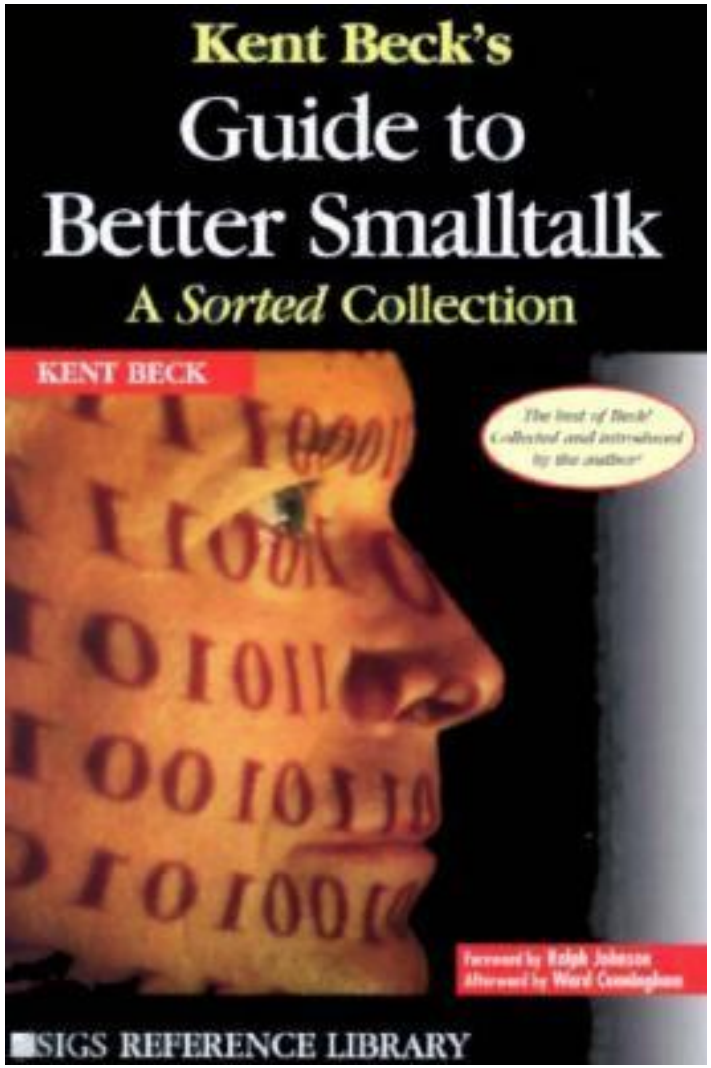
Kent Beck

- Author
  - Extreme Programming
- Original Signatory on the Agile Manifesto
- "Rediscovery" of TDD
  - Whaaaaaa?
- xUnit style of testing

SmallTalk


SMALLTALK-80
THE LANGUAGE AND ITS IMPLEMENTATION
Adele Goldberg and David Robson

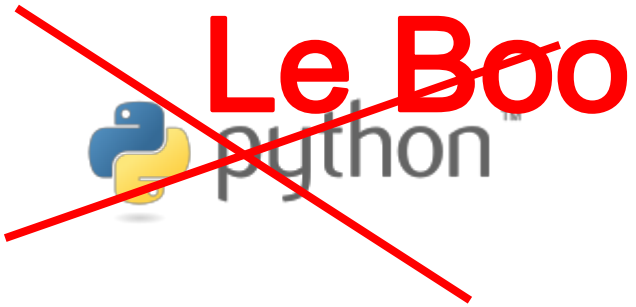
S-Unit

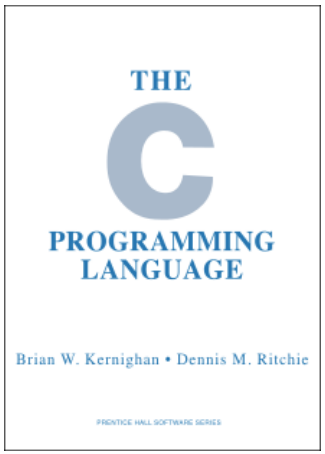

Kent Beck's
Guide to
Better Smalltalk
A Sorted Collection
KENT BECK
SIGS REFERENCE LIBRARY


Microsoft
.NET


php


Qt

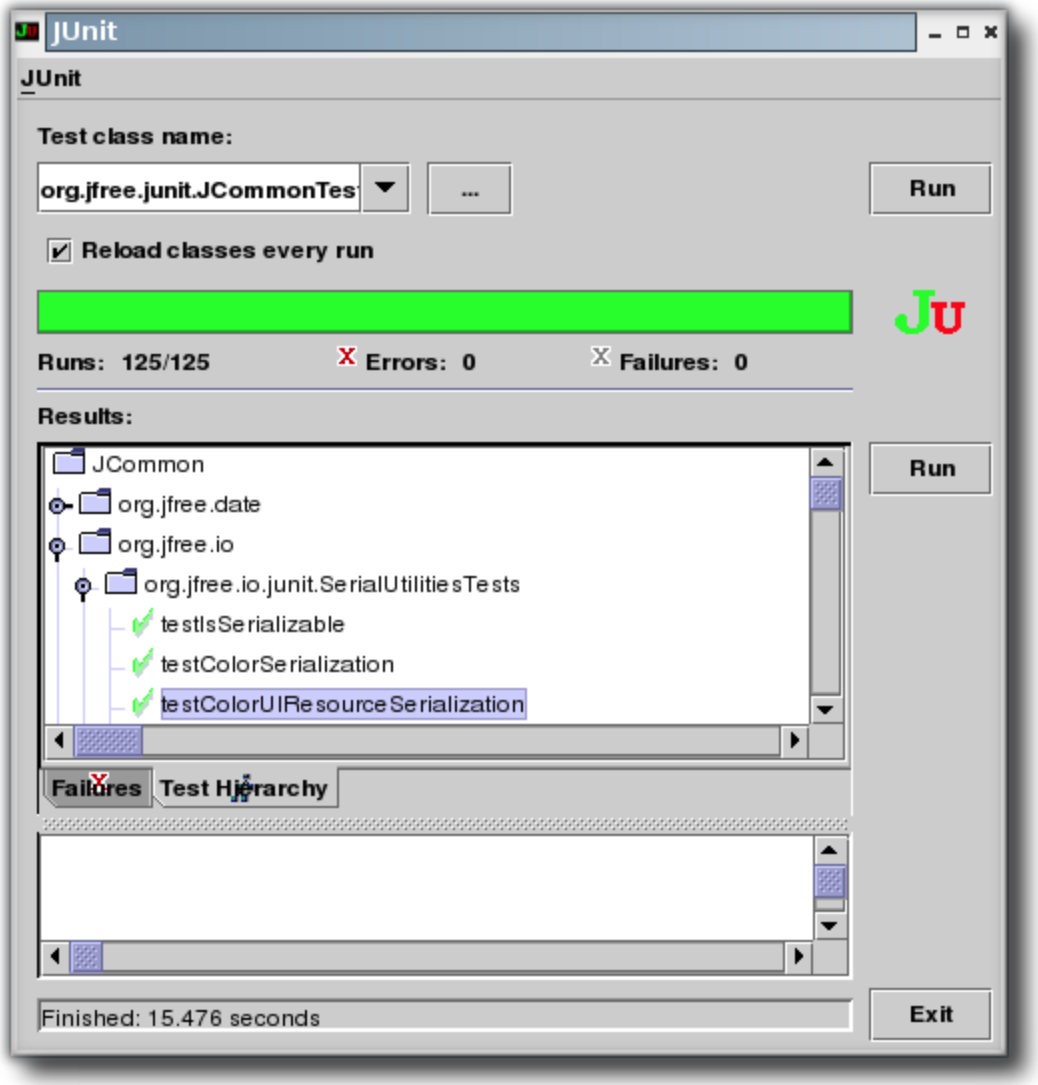

Le Boo
python


THE
C
PROGRAMMING
LANGUAGE
Brian W. Kernighan • Dennis M. Ritchie


PROGRAMMING REPUBLIC OF
PERL


Js


Java


JUnit
JUnit
Test class name:
org.jfree.junit.JCommonTes
...
Run
Reload classes every run
Runs: 125/125
Errors: 0
Failures: 0
Results:
JCommon
org.jfree.date
org.jfree.io
org.jfree.io.junit.SerialUtilitiesTests
testIsSerializable
testColorSerialization
testColorUIResourceSerialization
Run
Failures
Test Hierarchy
Finished: 15.476 seconds
Exit

Test::Unit

- The xUnit of Ruby
- Standard
- Dependable
- Simple

- Full Featured, xUnit
  - setup()
  - teardown()
  - test_case
  - fixtures
  - assertions

Test::Unit

test_unit.rb

```ruby
require './calculator'
require 'test/unit'

class TestSomething < Test::Unit::TestCase

  def setup
    # Prepare you environment for your tests
    @calc = Calculator.new
  end

  def test_case_1
    # Testing logic w/ assertions
    assert_equal @calc.divide(3,2), 1.5
  end

  # Mucho(Much) tests...

  def teardown
    # Cleanup your environment, before you wreck yourself
    @calc = nil
  end

end
```

Line: 20 Column: 16 Ruby on Rails Soft Tabs: 2 teardown

Test::Unit

What it does well

- Simple
- Easy fit
- Easily Combined

Test::Unit

Q: What testing framework do you use at 37signals? A: test/unit with the occasional splash of mocha. (That's all you need for great testing)

less than a minute ago via Tweetie for Mac

DHH
dhh

What it doesn't do so well

- Where to start in the process?
- What to test and what not to test?
- What to call the tests
- Relatability

Feature > Test > Code

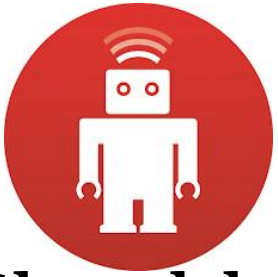
Welcome to the world Behavior Driven Development...

BDD, a history

- The first!
- Simple in theory, maybe
  - Given
  - When
  - Then
- Mapping requirements to code.
- The halo is not a mistake
  - Kind of a big deal

Flavors

Mini::Test



Shoulda

Cucumber behaviour driven development with elegance and joy

bacon

Riot

Cucumber behaviour driven development with elegance and joy

The truest implementation in Ruby

- RBehave Evolved
- Automated acceptance testing framework
- Maps your physical stories with a running test suite.
  - Sounds good!
  - So?
  - What does that mean?

Cucumber behaviour driven development with elegance and joy

Contrived code example!

cucumber.rb UNREGISTERED

```ruby
# Feature: Division
# * I have entered 3 into the calculator
# * I have entered 2 into the calculator
# * I press divide
# * the result should be 1.5 on the screen

Before do
  @calc = Calculator.new
end

After do
end

Given /I have entered (\d+) into the calculator/ do |n|
  @calc.push n.to_i
end

When /I press (\w+)/ do |op|
  @result = @calc.send op
end

Then /the result should be (.*) on the screen/ do |result|
  @result.should == result.to_f
end
```

Line 24, Column 4 Spaces: 2 Cucumber Steps

* Totally not taken from wikipedia

Cucumber behaviour driven development with elegance and joy

The problem with Cucumber

- Upfront implementation costs
- Unnatural hatred in community
- Can get convoluted
  - Sooooo... many.... defined steps!

A middle ground?

- Incorporates elements from xUnit and BDD
- Popular
  - Not really the new hotness anymore.
- Useful in all situations that cucumber and test/unit are
  - But don't necessarily conform to xUnit standard

King of the Hill(*currently)

- RSpec!
- Most in use framework for Rails
  - Therefore in Ruby
- Ridiculously supported
- Takes elements from both sides

Whoo! More contrived code!

rspec.rb UNREGISTERED

```ruby
require 'rubygems'
require 'rspec'

class Calculator

  # * I have entered 3 into the calculator
  # * I have entered 2 into the calculator
  # * I press divide
  # * the result should be 1.5 on the screen

  def divide(dividend, divisor)
    dividend/divisor.to_f
  end
end

describe Calculator do

  describe "Division" do
    it "should divide 3 by 2" do
      Calculator.new.divide(3,2).should eq(1.5)
    end
    # etc...
  end

end

```

Line 22, Column 13 Spaces: 2 RSpec

Drawbacks

- A little heavy...
- Syntax not as easily acceptable
  - Not xUnit, but close

Variations

Learn RSpec and you can do the others.

Mini::Test

Bacon

Riot

Why I Test

- I will make mistakes
  - Good!
- I hate going back to old code
  - Old code is ugly
- The end result is cleaner
  - Less code = Less to maintain
  - I like free time
- TDD takes less time than not
- I was challenged for proof
  - But it works... I swear it.

Prove it!

- Any time of day, any circumstance
  - Prove your code works
- Relate your code directly to requirements
  - Doesn't mean BDD-style, doesn't hurt though
- Fast, Fast, Fast
  - Test turnaround time is important
  - If it's not you won't do it
  - No one is going to wait for you

What do we test?

- Impossible to answer... sorry
- Every project is different
- Every project has different pain points
- Coverage doesn't count
  - Oooooo, edgy.

Steve Baker, an author of RSpec

- **"Test The Hard Stuff"**
- Don't stop testing parts because they're hard
  - Software is hard by nature, no excuses
- The untested parts are fragile, and may/*will break.

Question: Is it most likely to break in front of the user because it's JavaScript and JavaScript testing is hard, or is it most likely to break because you didn't test it?

Answer: Your users don't care.

testing_helpers.sample

Test::Unit
Rspec
Mini::Test
Bacon
Cucumber
seattlerb/heckle
parralel_tests
simple_cov
seattlerb/flog
test_bisect
roodi
square/cane
guard
metric_fu
jasmine
capybara

notahat/machinist
thoughtbot/factory_girl
freerange/mocha
VCR
chrisk/fakeweb
bblimke/webmock
seattlerb/flay
thumblemonks/riot
troessner/reek
autotest
nulldb
CI (Jenkins, Travis CI, etc...)
NewRelic (Performance)
metrical
shoulda
ffaker

ensure

end

Thanks everyone for coming out!

talk.kill

@bennett_stevens