

# Measuring Ruby code

Jonathan Dahl  
Slantwise Design

<http://slantwisedesign.com>

<http://railspikes.com>

test coverage

complexity

rcoy, heckle

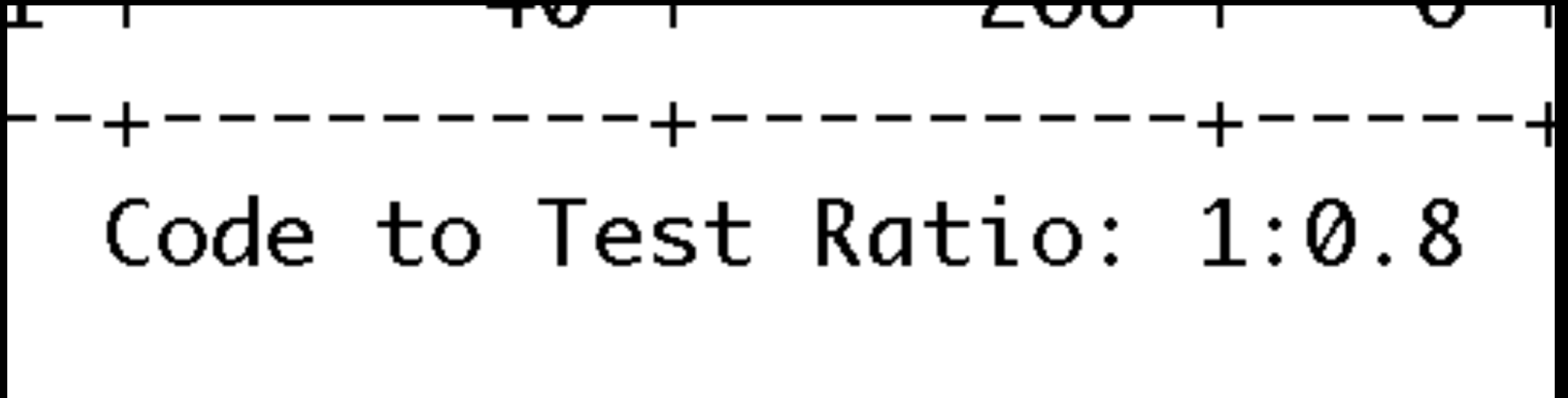
flog, saikuro

# Part I: Measuring Test Coverage

# rake stats

```
tk421:overlord jon$ rake stats
(in /Users/jon/projects/zencoder/overlord)
+-----+-----+-----+-----+-----+-----+
| Name           | Lines | LOC | Classes | Methods | M/C | LOC/M |
+-----+-----+-----+-----+-----+-----+
| Controllers    | 322   | 255 | 7        | 33       | 4    | 5      |
| Helpers        | 142   | 78  | 0        | 11       | 0    | 5      |
| Models         | 556   | 433 | 7        | 75       | 10   | 3      |
| Libraries      | 363   | 252 | 5        | 26       | 5    | 7      |
| Functional tests | 387   | 304 | 12       | 43       | 3    | 5      |
| Unit tests     | 622   | 509 | 9        | 80       | 8    | 4      |
+-----+-----+-----+-----+-----+-----+
| Total          | 2392  | 1831 | 40       | 268      | 6    | 4      |
+-----+-----+-----+-----+-----+-----+
Code LOC: 1018      Test LOC: 813      Code to Test Ratio: 1:0.8
```

# rake stats



Relative, not absolute, value. 1:1 doesn't necessarily mean anything.  
Medium: 1:0.8, High: 1:1.5

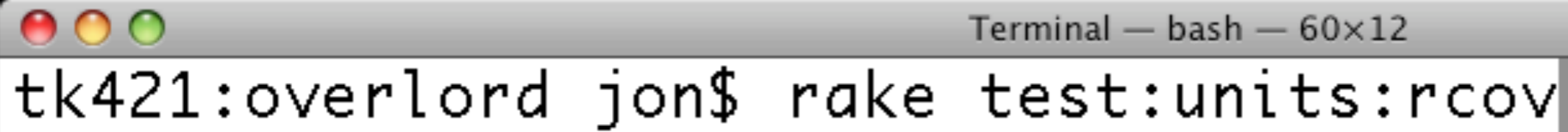
# RCov

runs your tests, and looks at what lines of application code were executed

# RCov

gem + Rails plugin

# RCov

A terminal window with a title bar that reads "Terminal — bash — 60x12". The window contains the command "tk421:overlord jon\$ rake test:units:rcov" followed by a grey cursor block.



```
tk421:overlord jon$ rake test:units:rcov
```

# RCov

```
Terminal — bash — 80x16
```

File	Lines	LOC	COV
lapp/controllers/application.rb	41	27	48.1%
lapp/helpers/application_helper.rb	42	33	51.5%
lapp/models/app.rb	40	30	100.0%
lapp/models/job.rb	324	249	80.7%
lapp/models/notifier.rb	37	30	93.3%
lapp/models/recipe.rb	17	15	73.3%
lapp/models/s3_account.rb	10	9	100.0%
lapp/models/uploaded_file.rb	42	33	100.0%
lapp/models/worker.rb	86	67	73.1%
llib/authentication.rb	72	50	90.0%
llib/spinoza_s3_backend.rb	151	79	69.6%
Total	862	622	78.9%

# RCov

Name	Total lines	Lines of code	Total coverage	Code coverage
<a href="#">app/models/recipe.rb</a>	17	15	76.5% 	73.3% 

```
1 class Recipe < ActiveRecord::Base
2   validates_presence_of :name, :command, :extension
3   validates_uniqueness_of :name
4   validates_format_of :name, :with => /\A(?:!zc_)/, :message => "cannot begin with 'z"
5   validates_format_of :name, :with => /\Azc /, :message => "must begin with 'zc_' if
6   validates_format_of :extension, :with => /\A[^\.]./, :message => "should not begin
7   #validates_format_of :extension, :with => /\A[^\.]./, :message => "contains invalid cha
8
9   class << self
10    def user_recipes
11      Recipe.find(:all, :conditions => "system_recipe = 0")
12    end
13    def system_recipes
14      Recipe.find(:all, :conditions => "system_recipe = 1")
15    end
16  end
17 end
```



# RCov

```
13 def system_recipes
14   Recipe.find(:all, :conditions => "system_recipe = 1")
15 end
```

# RCov

```
def test_system_recipes
  system_recipes = Recipe.system_recipes
  system_recipes.each do |sr|
    assert sr.system_recipe
  end
  other_recipes = Recipe.find(:all) - system_recipes
  other_recipes.each do |sr|
    assert !sr.system_recipe
  end
end
```

# RCov



Name	Total lines	Lines of code	Total coverage	Code coverage
<a href="#">app/models/recipe.rb</a>	17	15	100.0% 	100.0% 

```
1 class Recipe < ActiveRecord::Base
2   validates_presence_of :name, :command, :extension
3   validates_uniqueness_of :name
4   validates_format_of :name, :with => /\A(?:!zc_)/, :message => "cannot begin with 'zc_'"
5   validates_format_of :name, :with => /\Azc_/, :message => "must begin with 'zc_' if a s
6   validates_format_of :extension, :with => /\A[^\.]./, :message => "should not begin wit
7   #validates_format_of :extension, :with => /^[^]/, :message => "contains invalid charact
8
9   class << self
10    def user_recipes
11      Recipe.find(:all, :conditions => "system_recipe = 0")
12    end
13    def system_recipes
14      Recipe.find(:all, :conditions => "system_recipe = 1")
15    end
16  end
17 end
```

# RCov

```
def test_system_recipes
  system_recipes = Recipe.system_recipes
  # system_recipes.each do |sr|
  #   assert sr.system_recipe
  # end
  # other_recipes = Recipe.find(:all) - system_recipes
  # other_recipes.each do |sr|
  #   assert !sr.system_recipe
  # end
end
```

# RCov

Name	Total lines	Lines of code	Total coverage	Code coverage
<a href="#">app/models/recipe.rb</a>	17	15	100.0% 	100.0% 

```
1 class Recipe < ActiveRecord::Base
2   validates_presence_of :name, :command, :extension
3   validates_uniqueness_of :name
4   validates_format_of :name, :with => /\A(?:!zc_)/, :message => "cannot begin with 'zc_'"
5   validates_format_of :name, :with => /\Azc_/, :message => "must begin with 'zc_' if a s
6   validates_format_of :extension, :with => /\A[^\.]./, :message => "should not begin wit
7   #validates_format_of :extension, :with => /\A[^\s]$/, :message => "contains invalid charact
8
9   class << self
10    def user_recipes
11      Recipe.find(:all, :conditions => "system_recipe = 0")
12    end
13    def system_recipes
14      Recipe.find(:all, :conditions => "system_recipe = 1")
15    end
16  end
17 end
```

# RCov

Does NOT tell you what is tested...

# RCov

...only what is not tested.

# RCov

100% RCov coverage

100% test coverage

RCov

100% RCov coverage

100% test coverage

# Heckle

mutation tester, based on Jester (Java)

# Heckle

Your tests **SHOULD** fail  
if your code changes

# Heckle

```
tk421:overlord jon$ heckle Job assign_worker -t test/unit/job_test.rb  
Initial tests pass. Let's rumble.
```

```
*****  
*** Job#assign_worker loaded with 7 possible mutations  
*****
```

```
7 mutations remaining...  
6 mutations remaining...  
5 mutations remaining...  
4 mutations remaining...  
3 mutations remaining...  
2 mutations remaining...  
1 mutations remaining...
```

# Heckle

The following mutations didn't cause test failures:

```
--- original
+++ mutation
  def assign_worker(new_worker)
    self.worker = new_worker
-   self.started_at = Time.now.utc
+   self.started_at = nil.utc
    self.expires_at = 5.minutes.from_now.utc
    save!()
    worker.expire_others(id)
  end
```

# Heckle

```
--- original
+++ mutation
def assign_worker(new_worker)
  self.worker = new_worker
  self.started_at = Time.now.utc
- self.expires_at = 5.minutes.from_now.utc
+ self.expires_at = -61.minutes.from_now.utc
  save!()
  worker.expire_others(id)
end
```

# Heckle

```
--- original
+++ mutation
def assign_worker(new_worker)
  self.worker = new_worker
  self.started_at = Time.now.utc
  self.expires_at = 5.minutes.from_now.utc
  save!()
-  worker.expire_others(id)
end
```

# Heckle

```
def test_should_expire_other_jobs
  workers(:one).expects(:expire_others).with(jobs(:waiting).id)
  job = assign_worker_test
end
```

```
def test_should_timestamp_assign_worker_request
  job = assign_worker_test
  assert job.started_at
  assert job.expires_at
  assert job.expires_at > 4.minutes.from_now
  assert job.expires_at < 6.minutes.from_now
end
```

# Heckle

```
3 mutations remaining...  
2 mutations remaining...  
1 mutations remaining...  
No mutants survived. Cool!
```

Heckle Results:

```
Passed      :    1  
Failed      :    0  
Thick Skin:    0
```

All heckling was thwarted! YAY!!!

# 3 Heckle Gotchas

1. infinite loops

# 3 Heckle Gotchas

1. infinite loops

heckle Job -T 30

# 3 Heckle Gotchas

## 2. Coupled Code

# 3 Heckle Gotchas

## 2. Coupled Code

```
heckle Job -t test/unit/job.rb
```

# 3 Heckle Gotchas

## 3. Parent/included methods

# 3 Heckle Gotchas

3. Parent/included methods

heckle Job assign\_worker

# Part 2: Measuring Complexity

flog

# flog

code complexity

# flog

```
Job#status=: (14.8)  
  7.4: assignment  
  5.1: branch  
  4.4: utc  
  3.4: now  
  1.8: minutes  
  1.6: from_now  
  0.5: lit_fixnum
```

# flog

```
def status=(new_status)
  case new_status
  when 'Failed'
    self.failed_at = Time.now.utc
    self.expires_at = nil
  when 'Processing'
    self.expires_at = 5.minutes.from_now.utc
  when 'Processed'
    self.processed_at = Time.now.utc
    self.expires_at = nil
  end
end
```

# flog

```
tk421:overlord jon$ flog app/models/job.rb  
Total score = 456.15274670238
```

# flog

```
Job#stats_notified: (30.2)
```

```
def stats_notified(options = {})
  start = options.delete(:start) || 25.year.ago.utc
  finish = options.delete(:finish) || Time.now.utc.yesterday.at_midnight
  connection.select_value("SELECT
AVG(TIMESTAMPDIFF(SECOND,processed_at,notified_at)) FROM jobs WHERE
(notified_at IS NOT NULL AND failed_at IS NULL) and notified_at BETWEEN
'#{start.at_midnight.to_s(:db)}' AND '#{finish.to_s(:db)}'")
end
```

biased against time functions?

# flog

relative and arbitrary  
useful over time

saikuro

# saikuro

```
$ saikuro -c -i app/models/ -y 0
```

- c = cyclomatic complexity
- i = input path
- y = cc threshold for methods

# saikuro

**Class : Job**

**Total Complexity: 29**

**Total Lines: 175**

not necessarily useful  
except maybe ratio of complexity/methods

# saikuro

Method	Complexity	# Lines
assign_worker	1	6
recipe=	2	3
to_xml	1	5
duration_string	1	6
status	5	11
status=	4	11

# saikuro

a bit more objective than flog  
still probably more useful over time

# what to look for?

total – no  
average – probably not  
median + high scores  
change over time

# when to run?

autotest

growl

\*before scm checkin

\*continuous integration – GRAPH OVER TIME

\*when problems occur

# other uses

per sprint  
per developer

Worthwhile goal  
one indicator among many

Part 3:  
Jon Waxer Philosophical

# How can I write better code?

maximize test coverage

minimize complexity

why?

just because it is “right”?

pragmatic

# Thanks!

Jonathan Dahl  
Slantwise Design

<http://slantwisedesign.com>

<http://railspikes.com>