

Reducing RSpec Runtime by 4400%

by @jimmynguyc

- Engineering Team Lead @ RapidRiver (rrsoft.co)
- Specialized in Rails
- Remote First Teams
- We're Hiring!! :D

Outline

- What is RSpec
- What are Feature Specs
- What are scriptable, headless browsers
- Why is it so f**king slow
- How I fixed it

RSpec



Behaviour Driven
Development for Ruby.
Making TDD Productive and Fun.

```
# game_spec.rb

RSpec.describe Game do
  describe "#score" do
    it "returns 0 for an all gutter game" do
      game = Game.new
      20.times { game.roll(0) }
      expect(game.score).to eq(0)
    end
  end
end
```

```
$ rspec game_spec.rb --color --format doc
```

```
Game
  #score
    returns 0 for all gutter game
```

```
Finished in 0.00057 seconds
1 example, 0 failures
```

Feature Specs

- Feature specs are high-level tests meant to exercise **slices of functionality** through an application. They should drive the application only via its external interface, usually web pages.

```
require "rails_helper"
```

```
RSpec.feature "Widget management", :type => :feature do  
  scenario "User creates a new widget" do  
    visit "/widgets/new"
```

```
    fill_in "Name", :with => "My Widget"  
    click_button "Create Widget"
```

```
    expect(page).to have_text("Widget was successfully created.")  
  end  
end
```

- Pages are rendered
- Javascript executed
- CSS are applied
- Images are loaded

Headless & Scriptable Browser Engines

Browser Engines



WebKit
Open Source Web Browser Engine



KHTML

Popular RSpec HSBE

- capybara-webkit
- poltergeist with PhantomJS

Why Poltergeist?

- Both based on WebKit
- Better UI simulation (e.g. overlapping element prevent clicks)
- Can raise JS errors as test errors (js_errors: true)
- Easier to install (standalone executable vs qt5 dependencies)

Setup

Installation

Add this line to your Gemfile and run `bundle install`:

```
gem 'poltergeist'
```

In your test setup add:

```
require 'capybara/poltergeist'  
Capybara.javascript_driver = :poltergeist
```

<https://github.com/teampoltergeist/poltergeist>

Installing PhantomJS

You need at least PhantomJS 1.8.1. There are *no other external dependencies* (you don't need Qt, or a running X server, etc.)

Mac

- *Homebrew:* `brew install phantomjs`
- *MacPorts:* `sudo port install phantomjs`
- *Manual install:* [Download this](#)

Linux

- Download the [32 bit](#) or [64 bit](#) binary.
- Extract the tarball and copy `bin/phantomjs` into your `PATH`

Windows

- Download the [precompiled binary](#) for Windows

Manual compilation

Do this as a last resort if the binaries don't work for you. It will take quite a long time as it has to build WebKit.

- Download [the source tarball](#)
- Extract and cd in
- `./build.sh`

(See also the [PhantomJS building guide](#).)


<https://github.com/teampoltergeist/poltergeist>


But ...

Builds » EssayJack » ejb » bug-139542217-slow-specs » build 3141

 Rebuild



 Collecting test results failed!







































 Your build has exceeded the memory limit of 4G on 1 container. The results of this build are likely invalid. We have taken a snapshot of the memory usage at the time, which you can find in a build artifact named `memory-usage.txt`. The RSS column in this file shows the amount of memory used by each process, measured in kilobytes.

```
PID  RSS  %CPU  COMMAND
21419 3363472 18.6 /usr/local/bin/phantomjs --load-images=no --ignore-ssl-errors=yes --ssl-
20904 1002564 78.4 /home/ubuntu/ejb/vendor/bundle/ruby/2.3.0/bin/rspec --color --format doc
spec/features/subscriptions_spec.rb spec/controllers/papers_controller_spec.rb spec/models/
spec/presenters/essay_presenter_spec.rb spec/features/voucher_spec.rb
```


Memory leak

If you run a few capybara sessions manually please make sure you've called `session.driver.quit` when you don't need session anymore. Forgetting about this causes memory leakage and your system's resources can be exhausted earlier than you may expect.

<https://github.com/teampoltergeist/poltergeist#memory-leak>

 TIMED OUT  rebuild	#2688  Fix specs	 2 months ago  2:02:51  8c58394	1.0
 TIMED OUT  rebuild	#2683  Fix specs	 2 months ago  2:02:16  8c58394	1.0
 TIMED OUT  rebuild	#2681  Merge pull request #298 from EssayJack/karuna/feature-137330187-return-user-to-curre...	 2 months ago  2:02:40  47708ad	1.0
 FAILED  rebuild	#2665  Merge pull request #309 from EssayJack/karuna/feature-138317795-customize-empty-tabl...	 2 months ago  2:00:22  4ab88fb	1.0
 TIMED OUT  rebuild	#2663  fix specs	 2 months ago  2:03:40  8937e14	1.0
 TIMED OUT  rebuild	#2655  Merge branch 'bug-20170216-various-template-bugs'	 2 months ago  2:02:21  bcc7480	1.0
 TIMED OUT  rebuild	#2648  Merge branch 'bug-20170216-various-template-bugs'	 2 months ago  2:02:24  bcc7480	1.0

```
.....
command bundle exec rspec --color --require rails_helper --format RspecJunitFormatter --out /tmp/circle-j
unit/rspec/rspec.xml spec --format RSpec::Instafail took more than 120 minutes to run
```



alexey  CircleCI Employee

Nov '15

There is a hard limit of 120 minutes per command—this helps us ensure that the infrastructure can be redeployed fast enough in case of a security vulnerability while still not breaking your build and not leaving it in an inconsistent state.

Would it be possible to use multiple containers in parallel and split the tests / simulations between them? This should allow you to stay under the 2h limit.

Sorry for the inconvenience.

1 Reply 

Solution   

<https://discuss.circleci.com/t/timeout-more-than-120-minutes/640>

Why Slow AF?

Automatic test metadata collection

If you're using our inferred test steps for Ruby or Python then we'll automatically collect test metadata, though for RSpec, Minitest, and Django you'll need to do some configuration to to enable the formatters:

For RSpec:

Add this to your gemfile:

```
gem 'rspec_junit_formatter'
```

<https://circleci.com/docs/1.0/test-metadata/>

Builds » EssayJack » ejb » master » build 3128

Test Summary	Queue (1:20:22)	Debug via SSH	Artifacts	circle.
<div>▼ Container 0</div> <div><div>▼ \$CIRCLE_ARTIFACTS/</div><div><div>▼ rspec_test_reports/</div><div>rspec.xml</div></div></div> <div><div>▼ \$CIRCLE_TEST_REPORTS/</div><div>> rspec/</div></div> <div><div>▼ home/</div><div>> ubuntu/</div></div>				
<div>▼ Container 1</div> <div><div>▼ \$CIRCLE_TEST_REPORTS/</div><div><div>▼ rspec/</div><div>rspec.xml</div></div></div> <div><div>▼ home/</div><div>> ubuntu/</div></div>				

<https://github.com/benzittlau/junit-xml-parser>

Usage

- Place your xml files in the `input/` dir with `.xml` extensions.
- run `rake parse_results[<limit>]` to get the sorted results output to console as a table.
- run `rake parse_results_by_file[<limit>,<sort>]` to get the sorted results grouped by file output to console as a table.

DEMO

jimmy@MacBook-Pro: ~/Projects/junit-xml-parser master!

\$ bundle exec rake parse_results_by_file

Time	Average Time	Test Count	File
1739.62	20.23	86	./spec/features/editpage_static_structure_spec.rb
1042.85	11.99	87	./spec/features/editpage_dynamic_structure_spec.rb
680.3	15.46	44	./spec/features/templates_spec.rb
535.33	6.61	81	./spec/models/essay_spec.rb
503.06	6.37	79	./spec/models/template_spec.rb
477.28	9.01	53	./spec/models/tags_spec.rb
446.67	9.31	48	./spec/features/institution_manager/users_dashboard_spec.rb
391.41	2.92	134	./spec/controllers/essays_controller_spec.rb
358.94	12.38	29	./spec/features/essay_creation_spec.rb
346.72	7.88	44	./spec/features/logins_spec.rb
223.88	15.99	14	./spec/features/templates_sharing_spec.rb
198.71	14.19	14	./spec/features/essay_sharing_spec.rb
167.28	5.97	28	./spec/features/institution_manager/groups_edit_spec.rb
160.32	20.04	8	./spec/features/citations_spec.rb
145.17	5.81	25	./spec/features/subscriptions_spec.rb
142.38	7.91	18	./spec/features/institution_manager/groups_dashboard_spec.rb
133.9	2.31	58	./spec/controllers/papers_controller_spec.rb

DEBUGGING

↳ RSpec Steps

(or: why would I want to relearn how to write specs?)

RSpec Steps allows you to chain examples into a series of steps that run in sequence and which stop when a step fails. It's often incredibly useful to be able to assemble a series of tests that should all pass, but where completely isolating them is less than sensible.

Rationale

RSpec's philosophy is that all examples should be completely independent. This is a great philosophy for most purposes, and we recommend you stick to it in almost all cases. BUT, that complete separation of examples really sucks when you're trying to write long stories involving many requests. You are usually stuck with three choices:

1. Write a sequence of examples, each of which repeats the behavior of all previous examples. Downside: horrendously inefficient.
2. Write a single huge example which performs the entire story. Downside: only one description, no independent reporting of the steps of the story.
3. Use Cucumber. Downside: We agree totally with this guy: <http://bit.ly/dmXqnY>

RSpec-steps intentionally breaks RSpec's "independent" philosophy to let us get the only thing we really want from Cucumber - the ability to execute some examples in sequence, and skip subsequent steps after a failure.

<https://github.com/LRDesign/rspec-steps>

Disable reset of session in rspec #418

 Closed

kberridge opened this issue on Jul 19, 2011 · 1 comment



kberridge commented on Jul 19, 2011



By default capybara always resets the session after each test. But there are times when I would prefer that management of the session state was left up to me (ex: to avoid the need to login before every test).



joliss commented on Jul 26, 2011

Collaborator




Closing in favor of the pull request [#419](#) (which Jonas will have to take a look at). Thanks for sending the pull request!



 joliss closed this on Jul 26, 2011



 kberridge referenced this issue on Aug 30, 2011

Adds `reset_session_after_each` config option [#419](#)

 Closed



jnicklas commented on Aug 10, 2011

Collaborator



I'm not 100% convinced of this. It's a pretty uncommon usecase, and in case you really need it, you can hook Capybara into RSpec yourself, it's not a lot of code, and you can get full control over everything.

For the record, I solved the same problem by using token authentication, as opposed to username/password. On the first request, you just send the token and boom, you're logged in, no extra requests necessary. Sped up a test suite by almost 30%.

<https://github.com/teamcapybara/capybara/pull/419>

So Let's DIY

spec/support/capybara.rb

```
capbara.rb
1 # frozen_string_literal: true
2 module Capybara
3   class << self
4     alias ori_reset_sessions! reset_sessions!
5     def reset_sessions!
6       fetch_current_example = RSpec.respond_to?(:current_example) ? proc {
7         RSpec.current_example } : proc { |context| context.example }
8       example = fetch_current_example.call(self) || self.class
9       ori_reset_sessions! unless example.respond_to?(:metadata) &&
10         example.metadata[:skip_reset_session]
11     end
12   end
13 end
```

spec/rails_helper.rb

```
require 'capybara/rspec'~  
require 'support/capybara'~
```

```
~ config.append_after(:each) do~  
~   ~ example = fetch_current_example.call(self) || self.class~  
~   ~ unless example.respond_to?(:metadata) && example.metadata[:skip_reset_session]~  
~     ~ Capybara.reset_sessions!~  
~     ~ if defined?(page)~  
~       ~ page.driver.clear_network_traffic~  
~       ~ page.driver.restart~  
~     ~ end~  
~   ~ end~  
~ end~  
end~
```

spec/rails_helper.rb

```
def reset_session_after_all~  
  after(:all) do~  
    Capybara.reset_sessions!~  
    Warden.test_reset!~  
    DatabaseCleaner.clean~  
  end~  
end~
```

spec/my_spec.rb

```
describe 'Essay Edit Page Static', skip_reset_session: true do~  
  reset_session_after_all~
```

- change `before(:each)` to `before(:all)`
- turn `let()` variables to instance variables

THANKS