gem that

by James Adam

by James Adam!

(a constructive rant, and then gem-this)

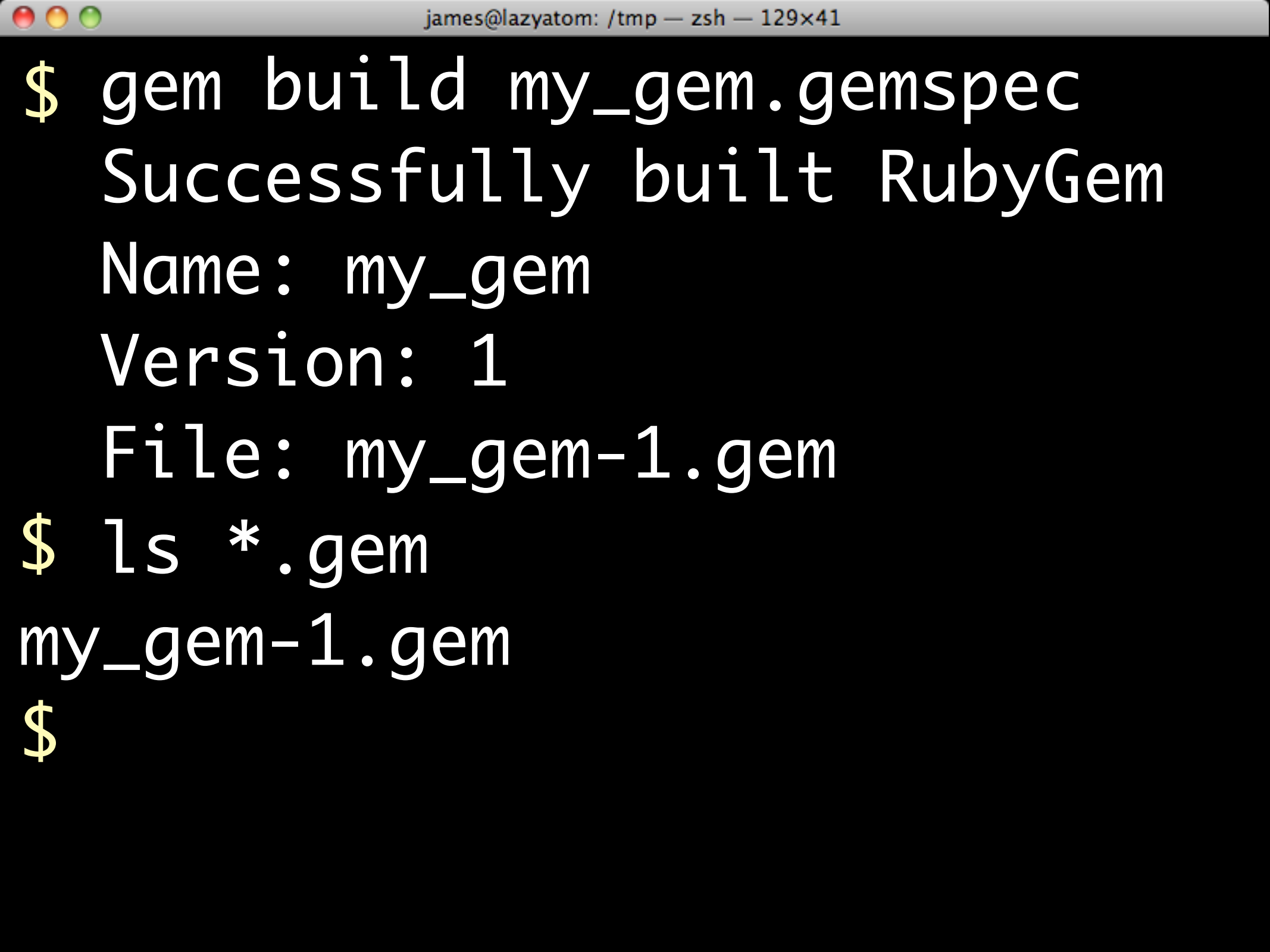
(and writing gem commands)

(by James Adam)

building gems - a primer

my_gem.gemspec

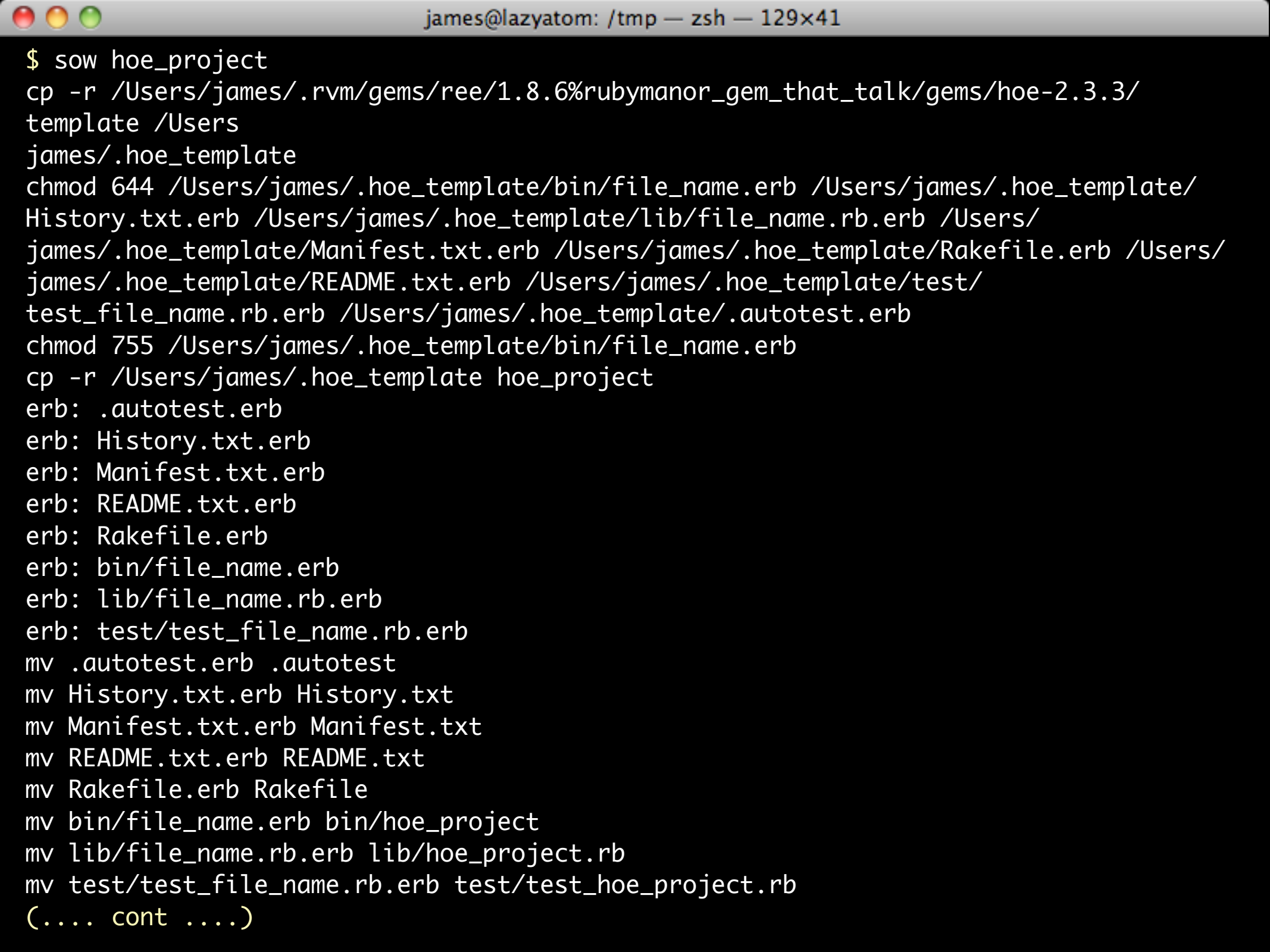
```ruby
Gem::Specification.new do |s|
  s.name = "my_gem"
  s.version = "1"

  s.authors = ["James Adam"]
  s.date = "2009-12-10"
  s.description = "What it does"
  s.email = "james@lazyatom.com"
  s.files = ["Rakefile", "lib/thing.rb"]
  s.homepage = "http://lazyatom.com"
  s.require_paths = ["lib"]

  # etc
end
```

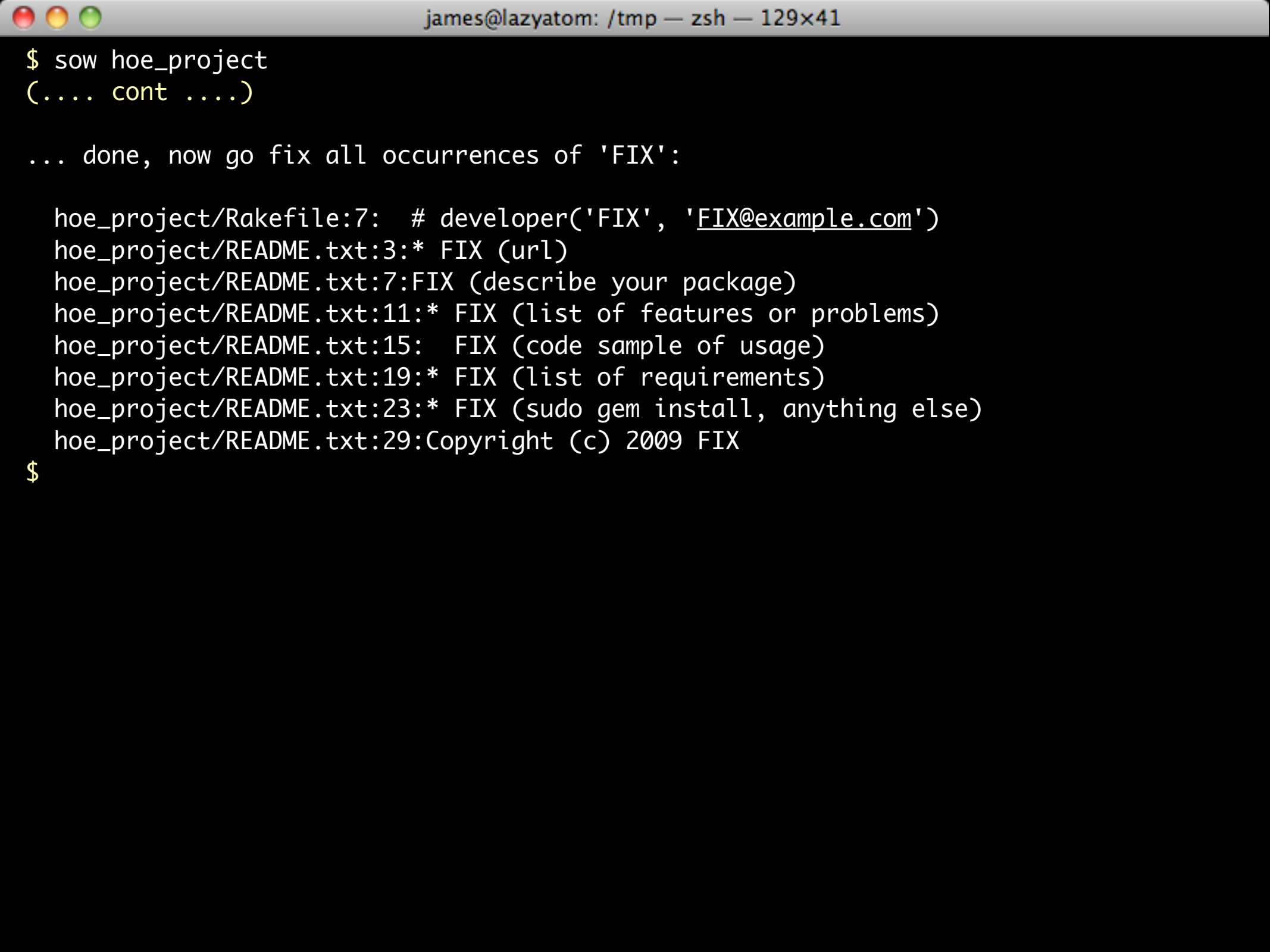

james@lazyatom: /tmp — zsh — 129×41
$ gem build my_gem.gemspec
Successfully built RubyGem
Name: my_gem
Version: 1
File: my_gem-1.gem
$ ls *.gem
my_gem-1.gem
$

my motivation

hoe

the ‘godfather’ of gem creation


james@lazyatom: /tmp — zsh — 129×41
$ sow hoe_project
cp -r /Users/james/.rvm/gems/ree/1.8.6%rubymanor_gem_that_talk/gems/hoe-2.3.3/
template /Users
james/.hoe_template
chmod 644 /Users/james/.hoe_template/bin/file_name.erb /Users/james/.hoe_template/
History.txt.erb /Users/james/.hoe_template/lib/file_name.rb.erb /Users/
james/.hoe_template/Manifest.txt.erb /Users/james/.hoe_template/Rakefile.erb /Users/
james/.hoe_template/README.txt.erb /Users/james/.hoe_template/test/
test_file_name.rb.erb /Users/james/.hoe_template/.autotest.erb
chmod 755 /Users/james/.hoe_template/bin/file_name.erb
cp -r /Users/james/.hoe_template hoe_project
erb: .autotest.erb
erb: History.txt.erb
erb: Manifest.txt.erb
erb: README.txt.erb
erb: Rakefile.erb
erb: bin/file_name.erb
erb: lib/file_name.rb.erb
erb: test/test_file_name.rb.erb
mv .autotest.erb .autotest
mv History.txt.erb History.txt
mv Manifest.txt.erb Manifest.txt
mv README.txt.erb README.txt
mv Rakefile.erb Rakefile
mv bin/file_name.erb bin/hoe_project
mv lib/file_name.rb.erb lib/hoe_project.rb
mv test/test_file_name.rb.erb test/test_hoe_project.rb
(.... cont ....)


james@lazyatom: /tmp — zsh — 129×41
$ sow hoe_project
(.... cont ....)
... done, now go fix all occurrences of 'FIX':
hoe_project/Rakefile:7: # developer('FIX', 'FIX@example.com')
hoe_project/README.txt:3:* FIX (url)
hoe_project/README.txt:7:FIX (describe your package)
hoe_project/README.txt:11:* FIX (list of features or problems)
hoe_project/README.txt:15: FIX (code sample of usage)
hoe_project/README.txt:19:* FIX (list of requirements)
hoe_project/README.txt:23:* FIX (sudo gem install, anything else)
hoe_project/README.txt:29:Copyright (c) 2009 FIX
$

hoe Rakefile

```ruby
# -*- ruby -*-
require 'rubygems'
require 'hoe'

Hoe.spec 'hoe_project' do
  # developer('FIX', 'FIX@example.com')

  # self.rubyforge_name = 'hoe_projectx'
  # if different than 'hoe_project'
end

# vim: syntax=ruby
```

hoe Rakefile

```
Hoe.spec 'hoe_project' do
    # ???
end
```


hoe
is a
fscking
virus


hoe
was a
fscking
virus

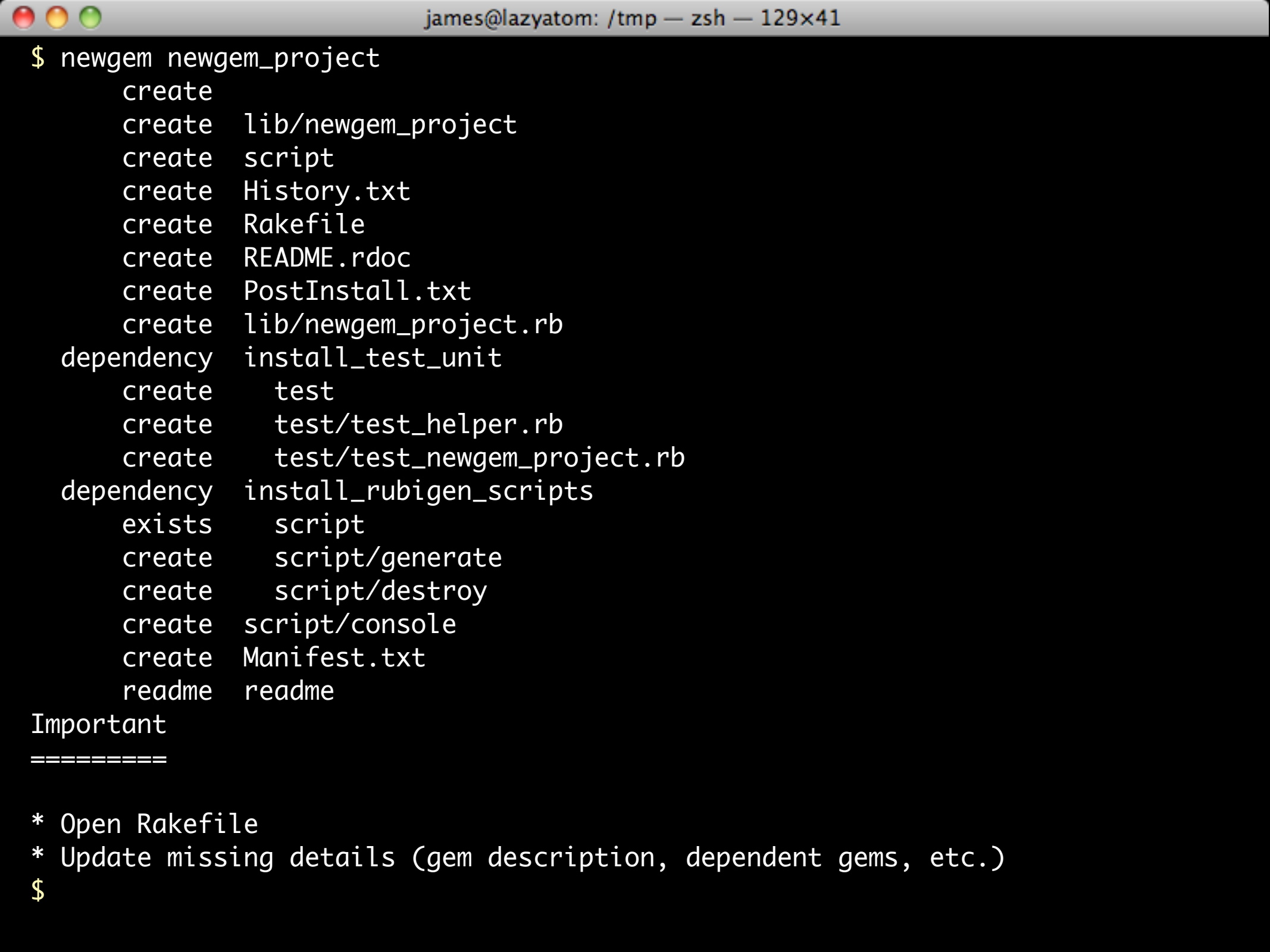
newgem

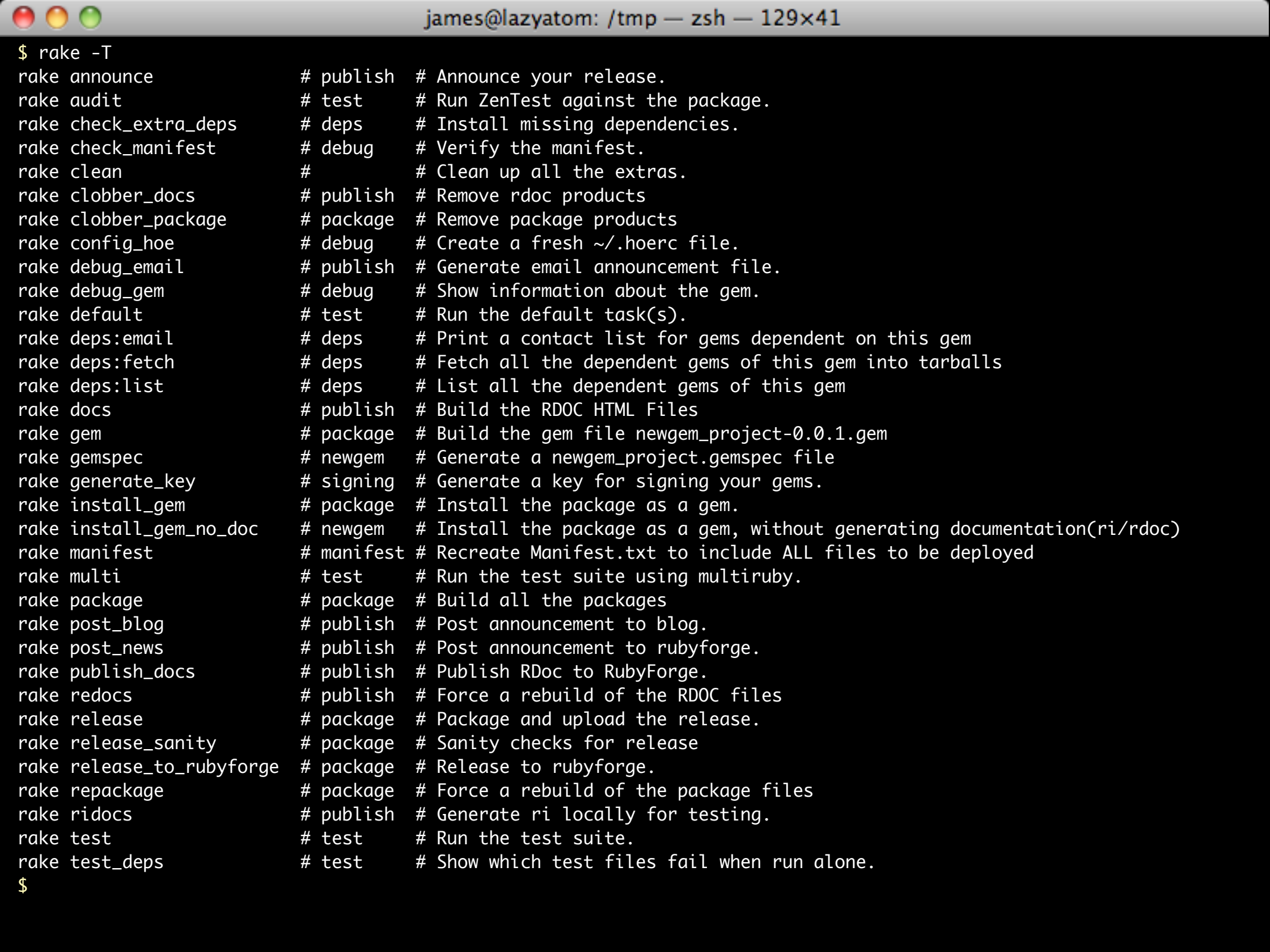
- Dr Nic

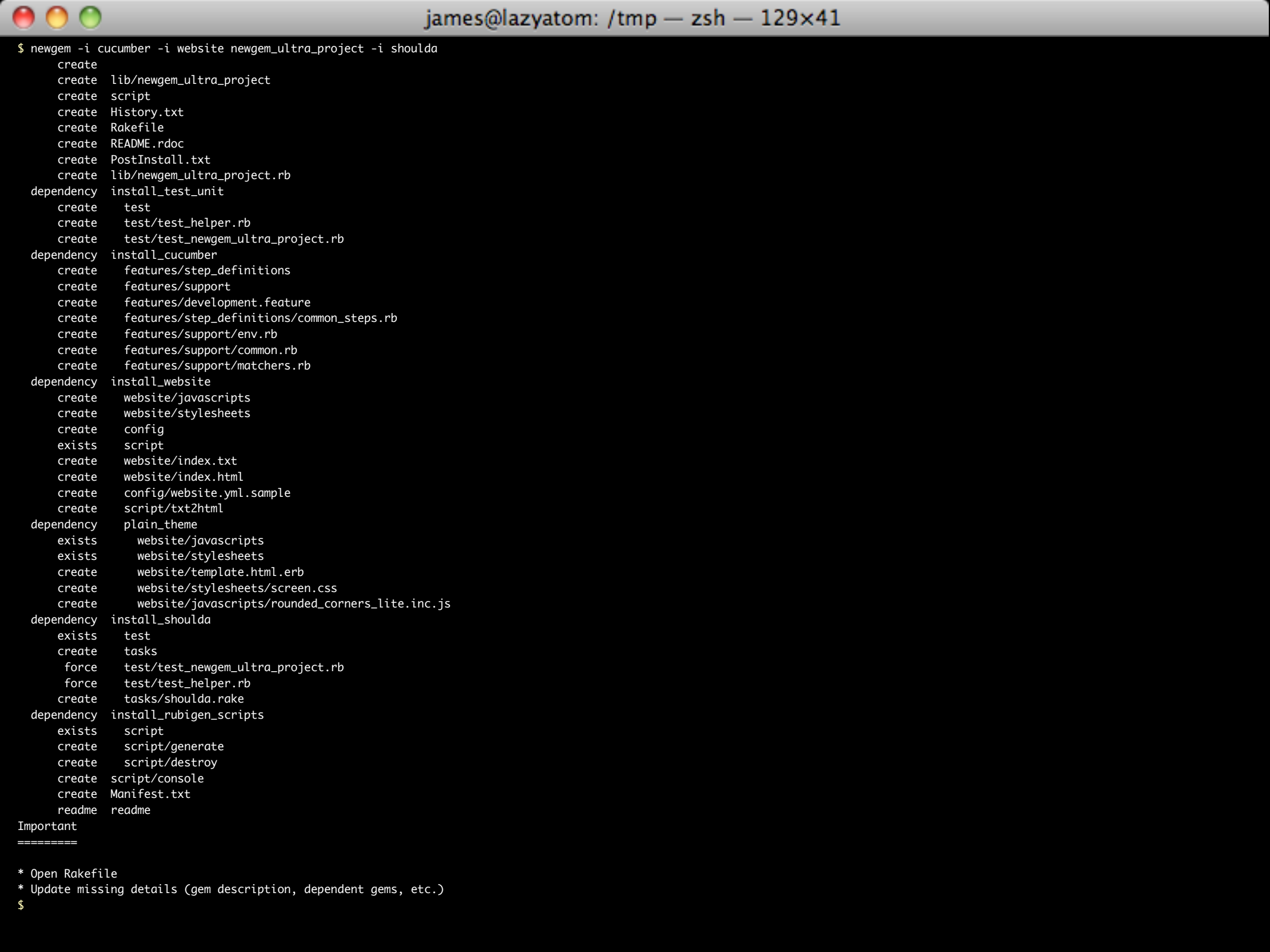
by Dr James Adam!

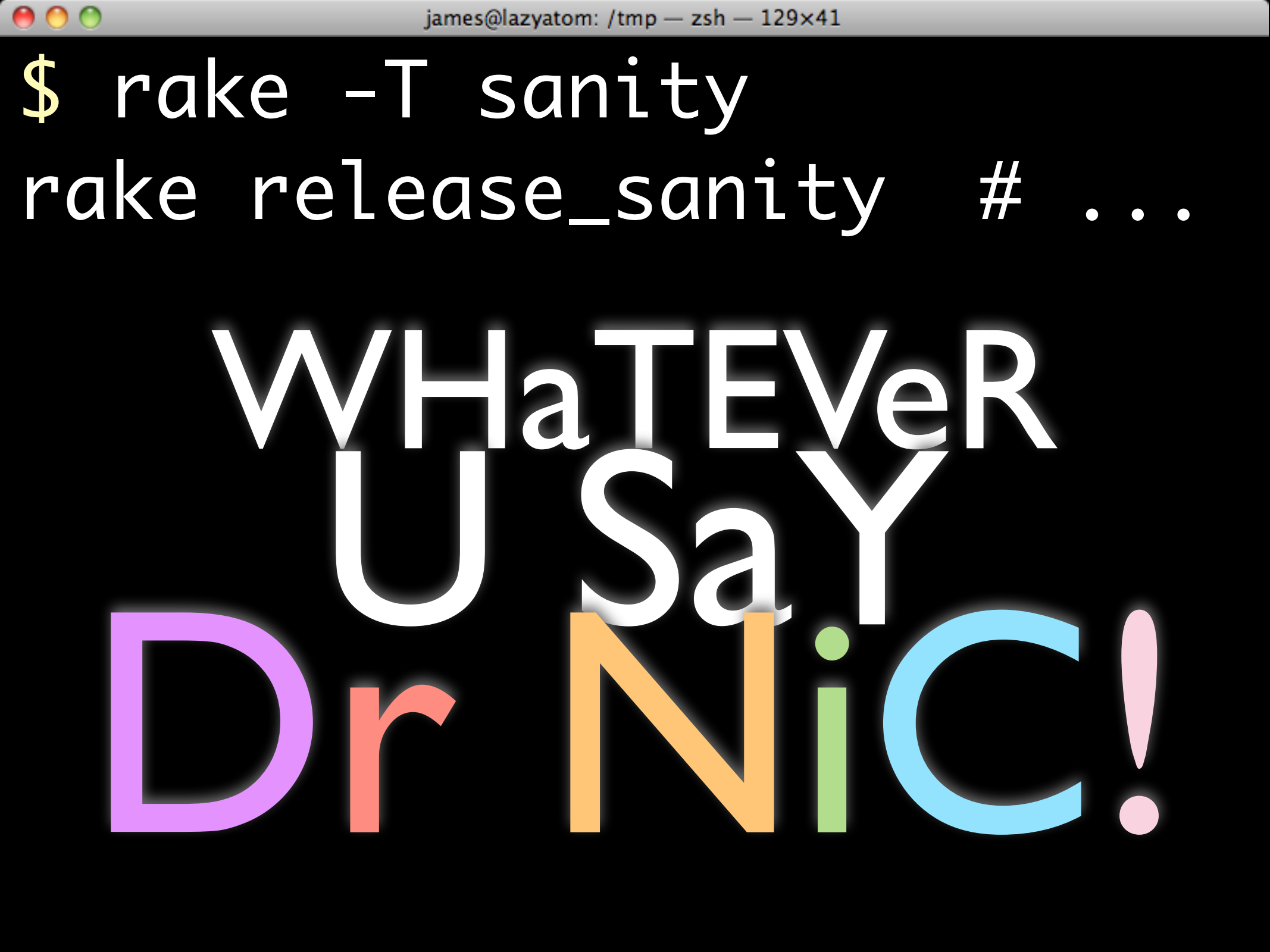
newgem

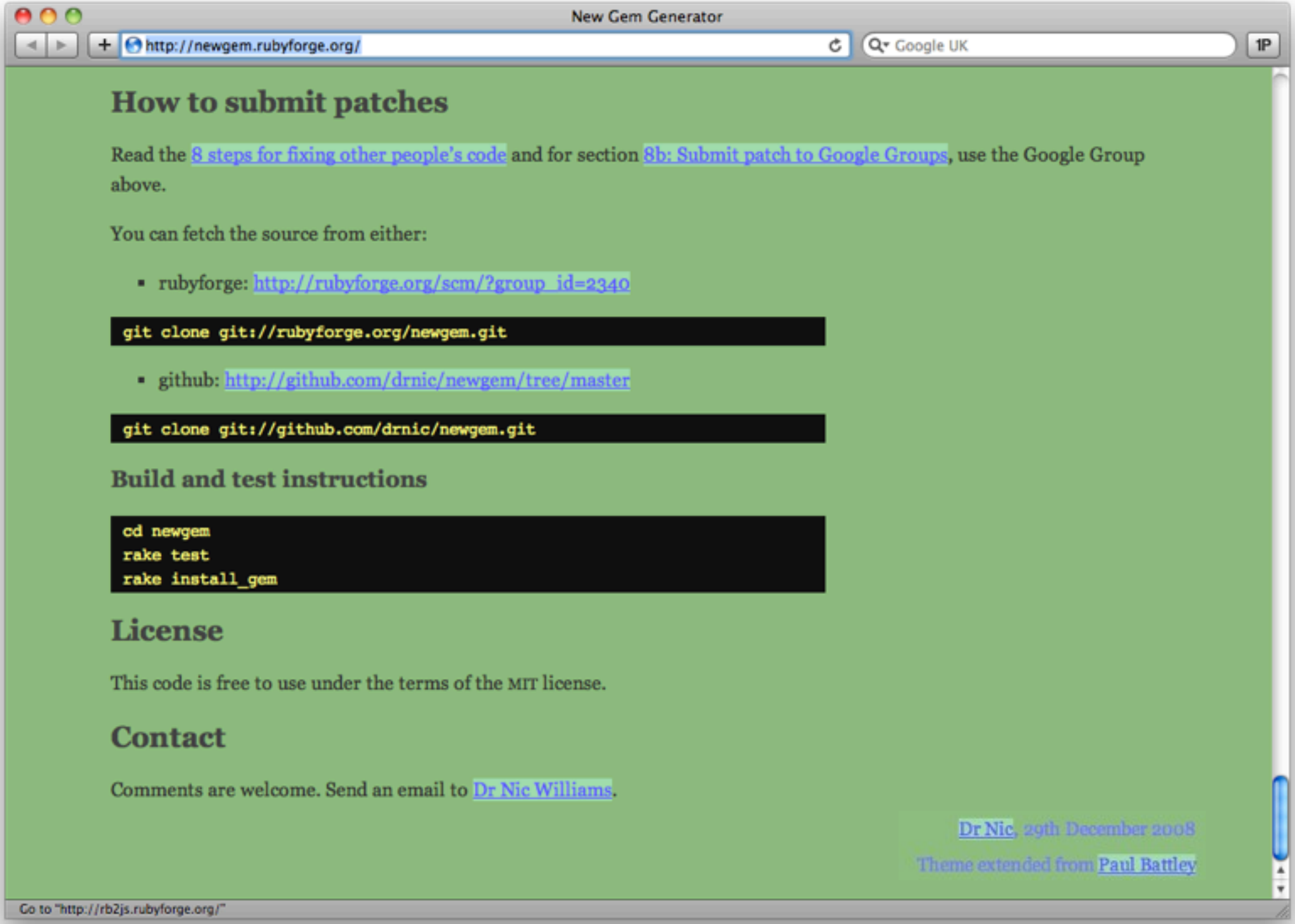
- generating structure, like the rails command
- extensible with other ‘generators’
- pretty bat-shit mental.


james@lazyatom: /tmp — zsh — 129×41
$ newgem newgem_project
create
create lib/newgem_project
create script
create History.txt
create Rakefile
create README.rdoc
create PostInstall.txt
create lib/newgem_project.rb
dependency install_test_unit
create test
create test/test_helper.rb
create test/test_newgem_project.rb
dependency install_rubigen_scripts
exists script
create script/generate
create script/destroy
create script/console
create Manifest.txt
readme readme
Important
=========
* Open Rakefile
* Update missing details (gem description, dependent gems, etc.)
$


james@lazyatom: /tmp — zsh — 129×41
$ rake -T
rake announce # publish # Announce your release.
rake audit # test # Run ZenTest against the package.
rake check_extra_deps # deps # Install missing dependencies.
rake check_manifest # debug # Verify the manifest.
rake clean # # Clean up all the extras.
rake clobber_docs # publish # Remove rdoc products
rake clobber_package # package # Remove package products
rake config_hoe # debug # Create a fresh ~/.hoerc file.
rake debug_email # publish # Generate email announcement file.
rake debug_gem # debug # Show information about the gem.
rake default # test # Run the default task(s).
rake deps:email # deps # Print a contact list for gems dependent on this gem
rake deps:fetch # deps # Fetch all the dependent gems of this gem into tarballs
rake deps:list # deps # List all the dependent gems of this gem
rake docs # publish # Build the RDOC HTML Files
rake gem # package # Build the gem file newgem_project-0.0.1.gem
rake gemspec # newgem # Generate a newgem_project.gemspec file
rake generate_key # signing # Generate a key for signing your gems.
rake install_gem # package # Install the package as a gem.
rake install_gem_no_doc # newgem # Install the package as a gem, without generating documentation(ri/rdoc)
rake manifest # manifest # Recreate Manifest.txt to include ALL files to be deployed
rake multi # test # Run the test suite using multiruby.
rake package # package # Build all the packages
rake post_blog # publish # Post announcement to blog.
rake post_news # publish # Post announcement to rubyforge.
rake publish_docs # publish # Publish RDoc to RubyForge.
rake redocs # publish # Force a rebuild of the RDOC files
rake release # package # Package and upload the release.
rake release_sanity # package # Sanity checks for release
rake release_to_rubyforge # package # Release to rubyforge.
rake repackage # package # Force a rebuild of the package files
rake ridocs # publish # Generate ri locally for testing.
rake test # test # Run the test suite.
rake test_deps # test # Show which test files fail when run alone.
$


james@lazyatom: /tmp — zsh — 129×41
$ newgem -i cucumber -i website newgem_ultra_project -i shoulda
create
create lib/newgem_ultra_project
create script
create History.txt
create Rakefile
create README.rdoc
create PostInstall.txt
create lib/newgem_ultra_project.rb
dependency install_test_unit
create test
create test/test_helper.rb
create test/test_newgem_ultra_project.rb
dependency install_cucumber
create features/step_definitions
create features/support
create features/development.feature
create features/step_definitions/common_steps.rb
create features/support/env.rb
create features/support/common.rb
create features/support/matchers.rb
dependency install_website
create website/javascripts
create website/stylesheets
create config
exists script
create website/index.txt
create website/index.html
create config/website.yml.sample
create script/txt2html
dependency plain_theme
exists website/javascripts
exists website/stylesheets
create website/template.html.erb
create website/stylesheets/screen.css
create website/javascripts/rounded_corners_lite.inc.js
dependency install_shoulda
exists test
create tasks
force test/test_newgem_ultra_project.rb
force test/test_helper.rb
create tasks/shoulda.rake
dependency install_rubigen_scripts
exists script
create script/generate
create script/destroy
create script/console
create Manifest.txt
readme readme
Important
=========
* Open Rakefile
* Update missing details (gem description, dependent gems, etc.)
$


james@lazyatom: /tmp — zsh — 129×41
$ rake -T sanity
rake release_sanity # ...
WHaTEVeR
U SaY
Dr NiC!


New Gem Generator
http://newgem.rubyforge.org/
Google UK
How to submit patches
Read the 8 steps for fixing other people's code and for section 8b: Submit patch to Google Groups, use the Google Group above.
You can fetch the source from either:
rubyforge: http://rubyforge.org/scm/?group_id=2340
git clone git://rubyforge.org/newgem.git
github: http://github.com/drnic/newgem/tree/master
git clone git://github.com/drnic/newgem.git
Build and test instructions
cd newgem
rake test
rake install_gem
License
This code is free to use under the terms of the MIT license.
Contact
Comments are welcome. Send an email to Dr Nic Williams.
Dr Nic, 29th December 2008
Theme extended from Paul Battley
Go to "http://rb2js.rubyforge.org/"


Dr Nic, 29th December 2008
Theme extended from Paul Battley

echoe

= (hoe) - (it being a dependency)

echoe Rakefile

```
require 'echoe'
Echoe.new('gem_name')
```

echoe Rakefile

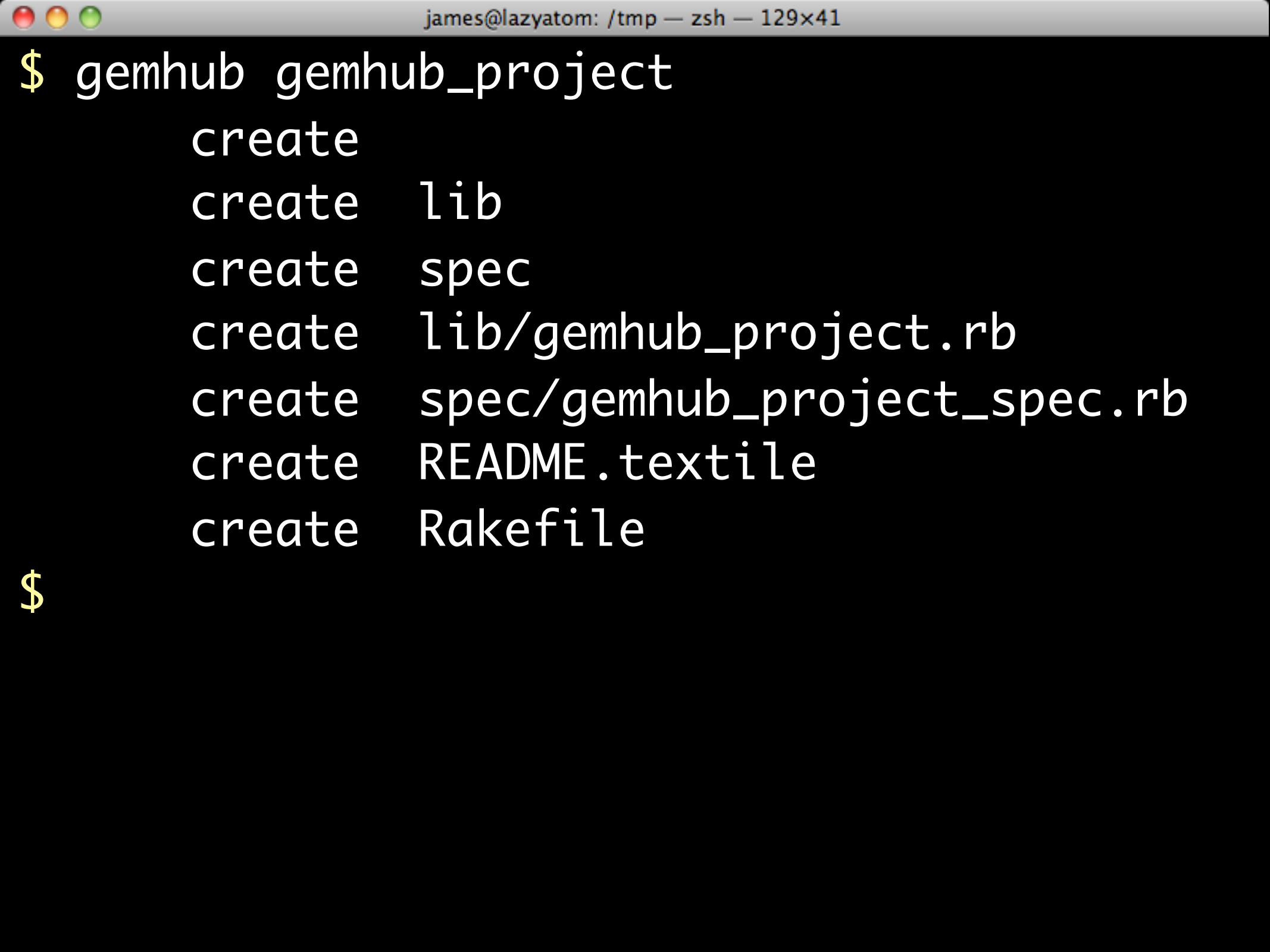
```ruby
Echoe.new("vanilla.rb") do |p|
  p.author = "James Adam"
  p.summary = "A talk about this would've
               been awesome"
  p.url = "http://interblah.net"
  p.runtime_dependencies = ["soup >=1.9.9"]
  # etc... ?
end
```

echoe Rakefile

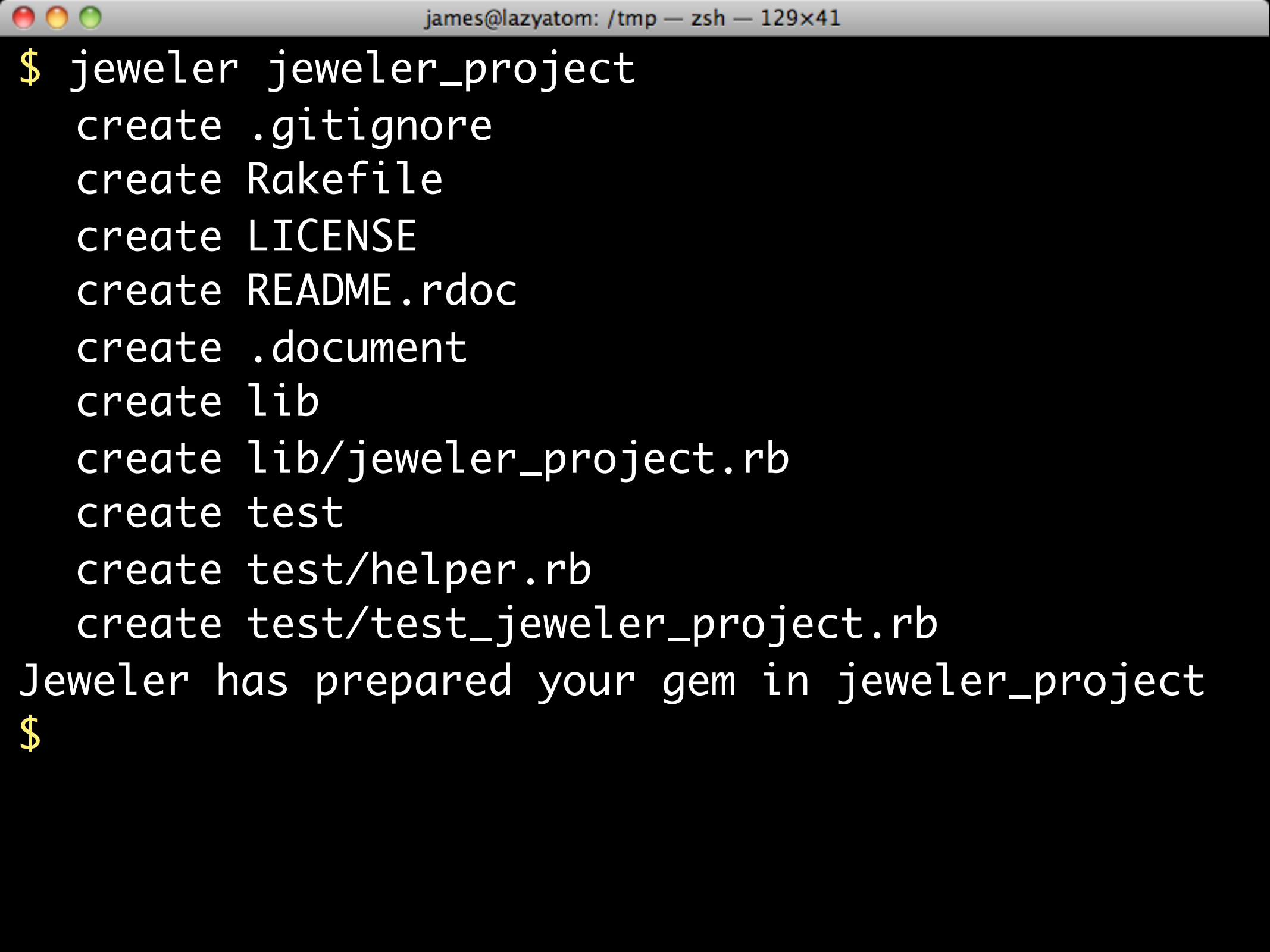
```ruby
begin
  require 'echoe'
  Echoe.new('my_gem')
rescue LoadError
  # probably nothing
end
# the rest of your Rakefile
```

gemhub

(somewhat of a mystery to me)

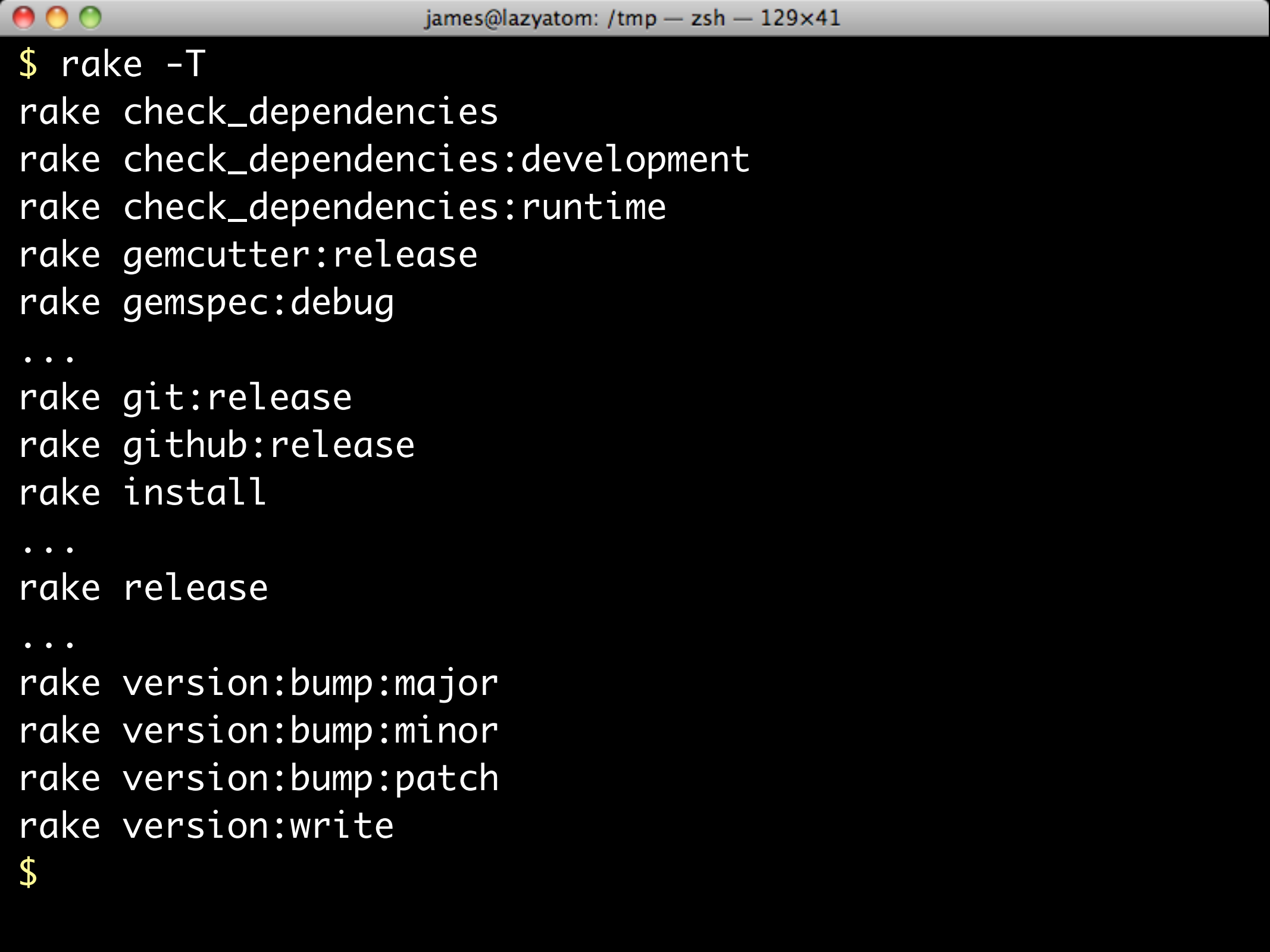

james@lazyatom: /tmp — zsh — 129×41
$ gemhub gemhub_project
create
create lib
create spec
create lib/gemhub_project.rb
create spec/gemhub_project_spec.rb
create README.textile
create Rakefile
$


jeweler
the choice of a new generation


james@lazyatom: /tmp — zsh — 129×41
$ jeweler jeweler_project
create .gitignore
create Rakefile
create LICENSE
create README.rdoc
create .document
create lib
create lib/jeweler_project.rb
create test
create test/helper.rb
create test/test_jeweler_project.rb
Jeweler has prepared your gem in jeweler_project
$

jeweler Rakefile

```ruby
begin
  require 'jeweler'
  Jeweler::Tasks.new do |gem|
    gem.name = "jeweler_project"
    gem.summary = %Q{summary of your gem}
    gem.email = "james@lazyatom.com"
    gem.homepage = "http://github.com/..."
    gem.authors = ["James Adam"]
    # etc ...
  end
  Jeweler::GemcutterTasks.new
rescue LoadError
  puts "Jeweler not available...."
end
```


james@lazyatom: /tmp — zsh — 129×41
$ rake -T
rake check_dependencies
rake check_dependencies:development
rake check_dependencies:runtime
rake gemcutter:release
rake gemspec:debug
...
rake git:release
rake github:release
rake install
...
rake release
...
rake version:bump:major
rake version:bump:minor
rake version:bump:patch
rake version:write
$

jeweler

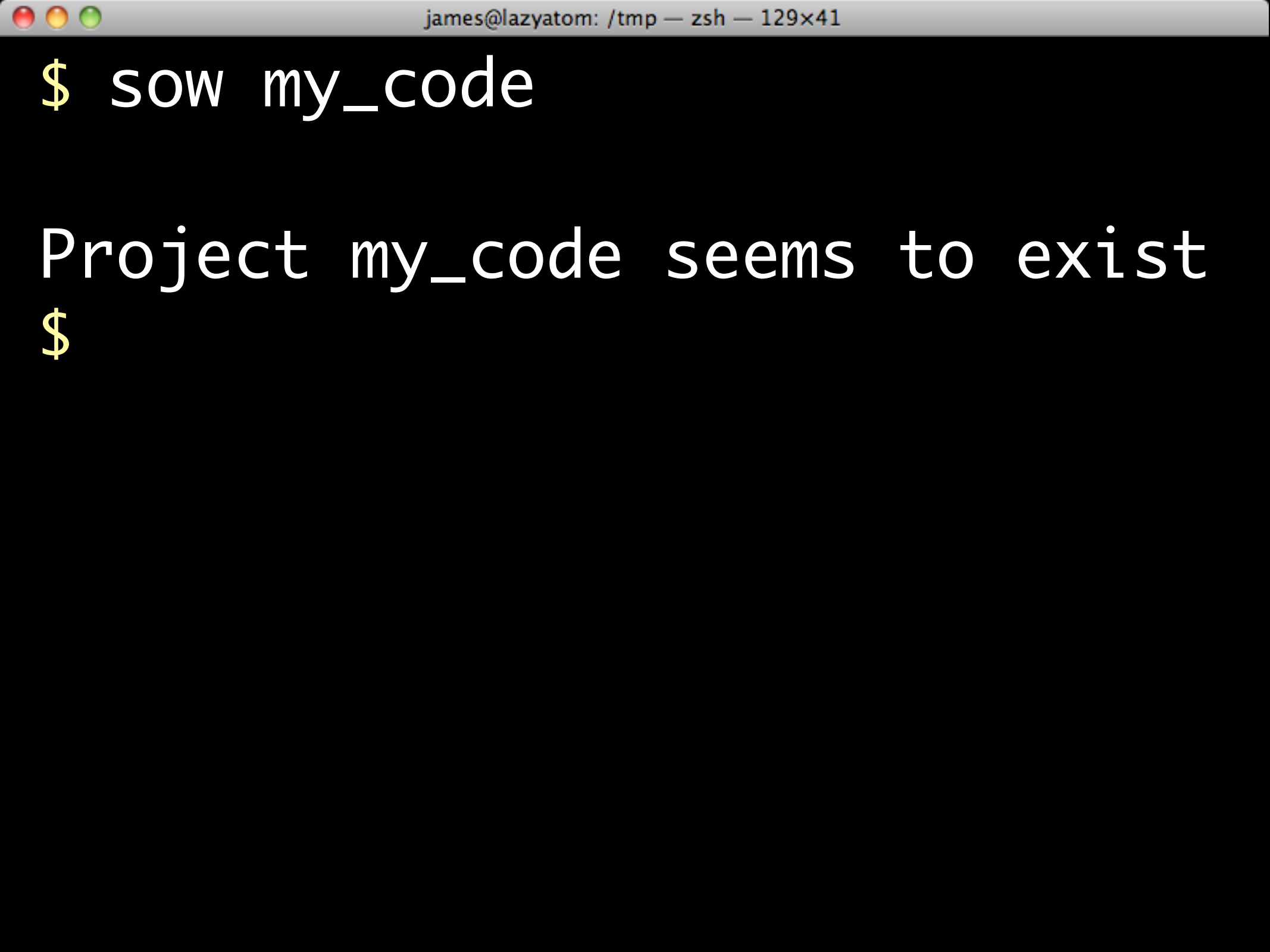
- zeitgeisty
- tailored for github, apparently
- still provides tasks for bumping versions, publishing
- still provides a generator

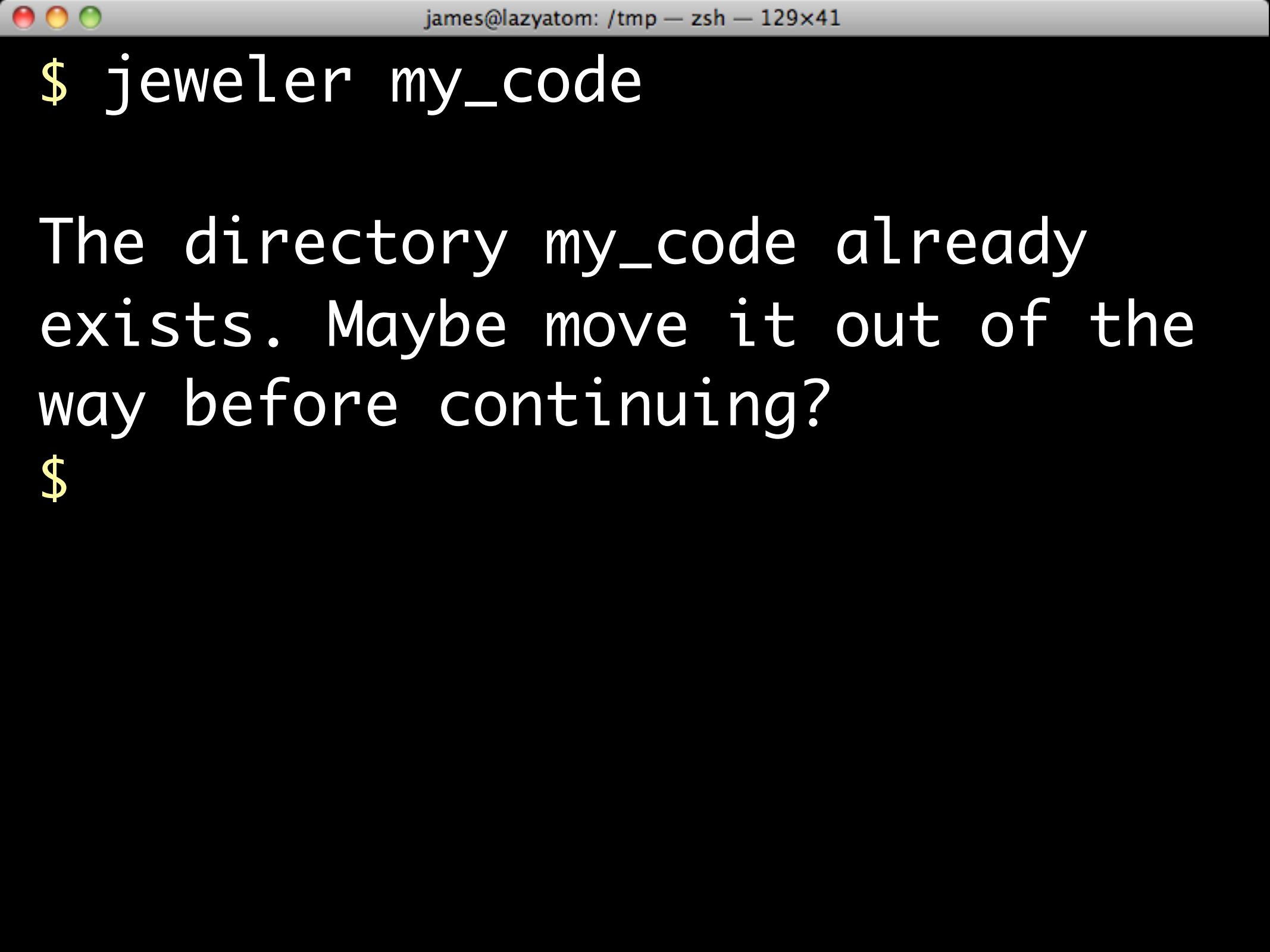
too opinionated

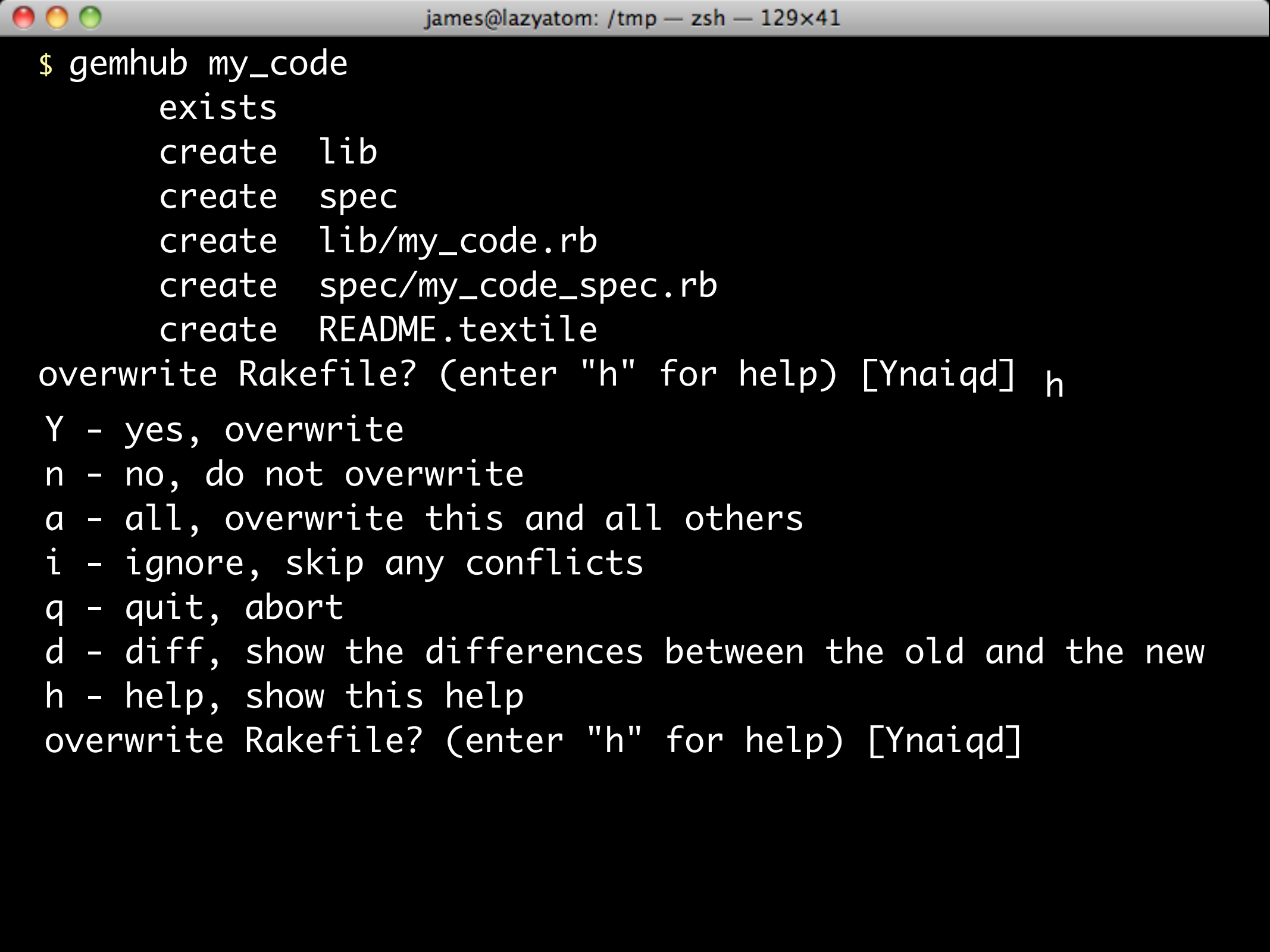
- don't need a tool to manage versions, websites, release notes, etc
- don't need a generator
- why so opaque?

what I want

- turn existing code into a gem
- get the fuck out of my way


james@lazyatom: /tmp — zsh — 129×41
$ sow my_code
Project my_code seems to exist
$


james@lazyatom: /tmp — zsh — 129×41
$ jeweler my_code
The directory my_code already exists. Maybe move it out of the way before continuing?
$


james@lazyatom: /tmp — zsh — 129×41
$ gemhub my_code
exists
create lib
create spec
create lib/my_code.rb
create spec/my_code_spec.rb
create README.textile
overwrite Rakefile? (enter "h" for help) [Ynaiqd] h
Y - yes, overwrite
n - no, do not overwrite
a - all, overwrite this and all others
i - ignore, skip any conflicts
q - quit, abort
d - diff, show the differences between the old and the new
h - help, show this help
overwrite Rakefile? (enter "h" for help) [Ynaiqd]

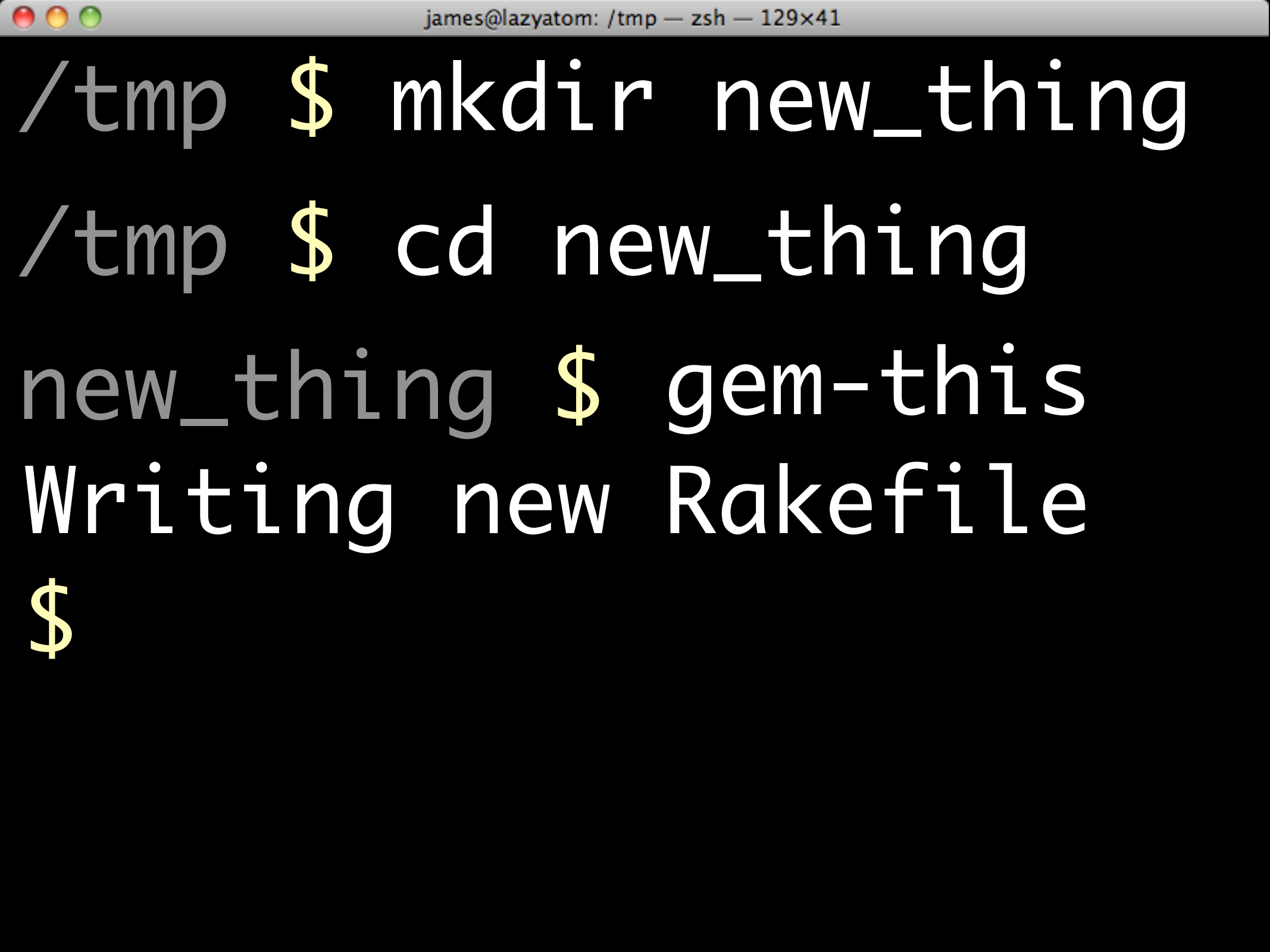
GEM THIS

gem this

- produces a simple Rakefile
- builds your gem
- maybe does your docs
- release to rubyforge

gem this

- produces a simple Rakefile
- builds your gem
- maybe does your docs
- ~~release to rubyforge~~

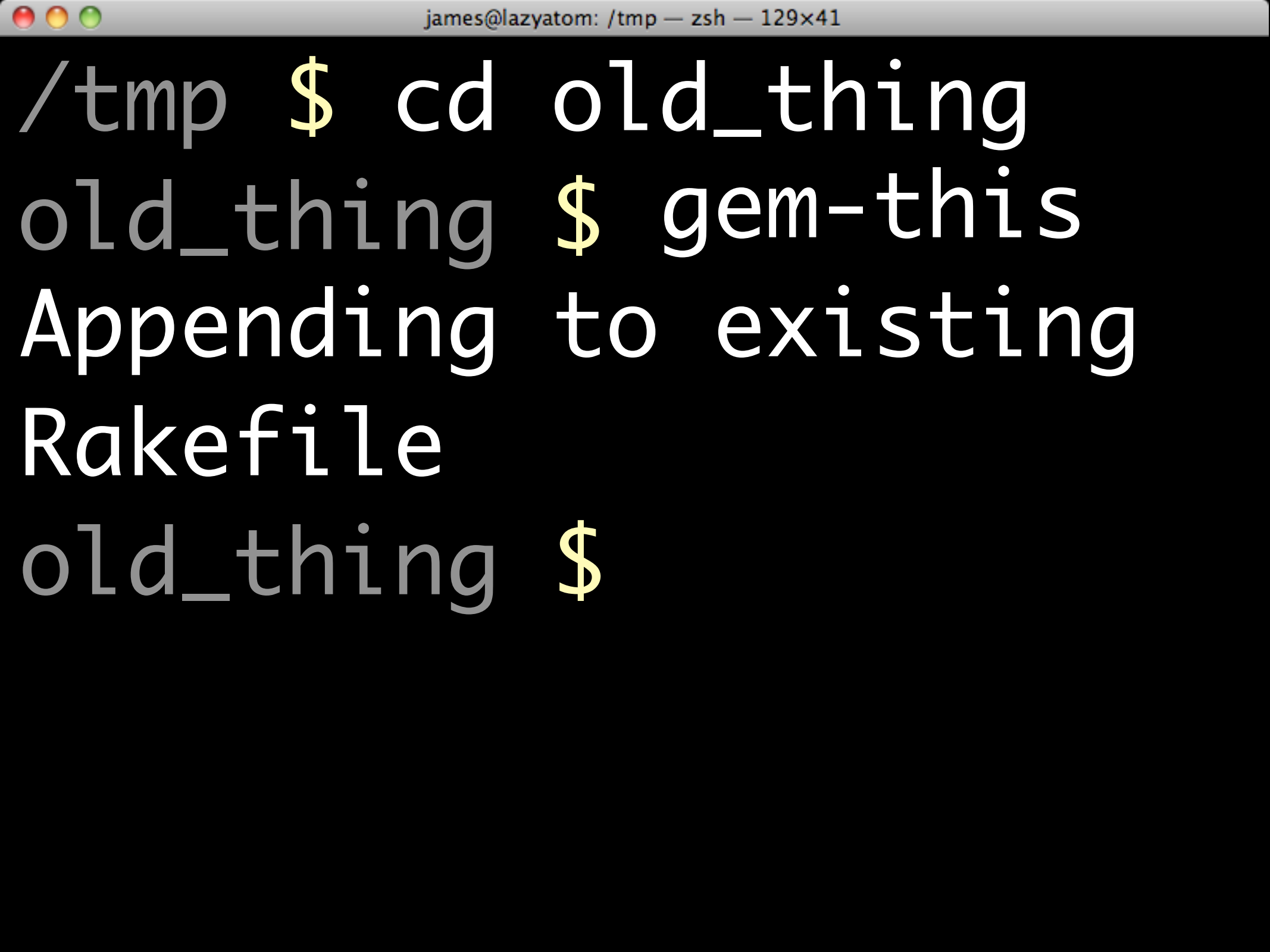

james@lazyatom: /tmp — zsh — 129×41
/tmp $ mkdir new_thing
/tmp $ cd new_thing
new_thing $ gem-this
Writing new Rakefile
$

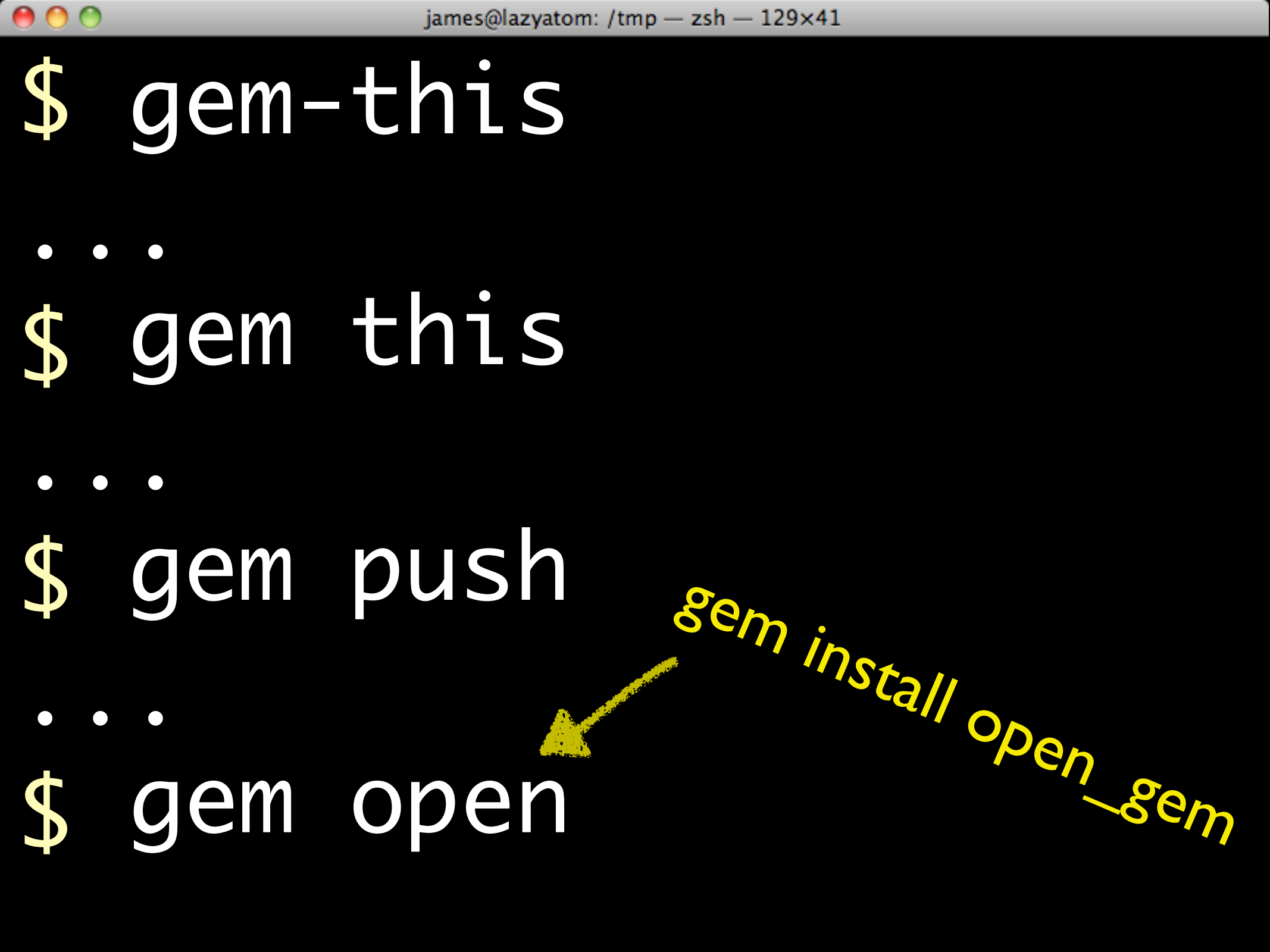
gem this Rakefile

```ruby
# This builds the actual gem. For details of what
all these options mean, and other ones you can
add, check the documentation here:
#
#   http://rubygems.org/read/chapter/20
#
spec = Gem::Specification.new do |s|

  # Change these as appropriate
  s.name              = "existing_project"
  s.version           = "0.1.0"
  s.summary           = "What this thing does"
  s.author            = "James Adam"
  # etc...
```

gem this Rakefile

```ruby
# This task actually builds the gem.
Rake::GemPackageTask.new(spec) do |pkg|
  pkg.gem_spec = spec
end

# Generate documentation
Rake::RDocTask.new do |rd|
  rd.rdoc_files.include("lib/**/*.rb")
  rd.rdoc_dir = "rdoc"
end
```


james@lazyatom: /tmp — zsh — 129×41
/tmp $ cd old_thing
old_thing $ gem-this
Appending to existing
Rakefile
old_thing $


james@lazyatom: /tmp — zsh — 129×41
$ gem-this
...
$ gem this
...
$ gem push
...
$ gem open
gem install open_gem

gem commands

the secret sauce

- subclass Gem::Command ...
- ... in a file called rubygems_plugin.rb ...
- ... and ensure it's in the $LOAD_PATH

my_gem/lib/rubygems_plugin.rb

```ruby
require 'rubygems/command_manager'
require 'rubygems/command'

class TestCommand < Gem::Command
  def initialize
  end

  def summary
  end

  def execute
  end
end
```

my_gem/lib/rubygems_plugin.rb

```ruby
require 'rubygems/command_manager'
require 'rubygems/command'

class TestCommand < Gem::Command
  def initialize
  end

  def summary
  end

  def execute
  end
end
```

my_gem/lib/rubygems_plugin.rb

```ruby
class TestCommand < Gem::Command
  def initialize
    super 'name', 'short description', {:debug => false}
    add_option('-d', '--debug', 'guess!') do |d, options|
      options[:debug] = d
    end
  end

  # def summary
  # def execute
end
```

my_gem/lib/rubygems_plugin.rb

```ruby
class TestCommand < Gem::Command
  def initialize
    super 'name', 'short description', {:debug => false}
    add_option('-d', '--debug', 'guess!') do |d, options|
      options[:debug] = d
    end
  end

  # def summary
  # def execute
end
```

my_gem/lib/rubygems_plugin.rb

```ruby
class TestCommand < Gem::Command
  def initialize
    super 'name', 'short description', {:debug => false}
    add_option('-d', '--debug', 'guess!') do |d, options|
      options[:debug] = d
    end
  end

  # def summary
  # def execute
end
```

my_gem/lib/rubygems_plugin.rb

```ruby
class TestCommand < Gem::Command
  # def initialize

  def summary
    "What this command does..."
  end

  # def execute
end
```

my_gem/lib/rubygems_plugin.rb

```ruby
class TestCommand < Gem::Command
  # def initialize

  # def summary

  def execute
    puts "You ran me with
          #{options.inspect}"
  end
end
```

my_gem/lib/rubygems_plugin.rb

```ruby
class TestCommand < Gem::Command
  # def initialize

  # def summary

  def execute
    puts "You ran me with
          #{options.inspect}"
  end
end
```

my_gem/lib/rubygems_plugin.rb

```ruby
# the command class
# ...

Gem::CommandManager.instance.
        register_command :test
```

gem_this/lib/rubygems_plugin.rb

```ruby
%w(rubygems/command_manager rubygems/command
   gem_this).each { |lib| require lib }

class ThisCommand < Gem::Command
  def initialize
    super 'this', GemThis::SUMMARY, :debug => false
    add_option('-d', '--debug', GemThis::DEBUG_MESSAGE) do |d, options|
      options[:debug] = d
    end
  end

  def summary; GemThis::SUMMARY; end

  def execute
    GemThis.new(File.basename(Dir.pwd), options[:debug]).create_rakefile
  end
end

Gem::CommandManager.instance.register_command :this
```

FUCK
YOUR
CONVENTIONS

james@lazyatom.com

github.com/lazyatom/gem-this

gem install gem-this

http://gofreerange.com