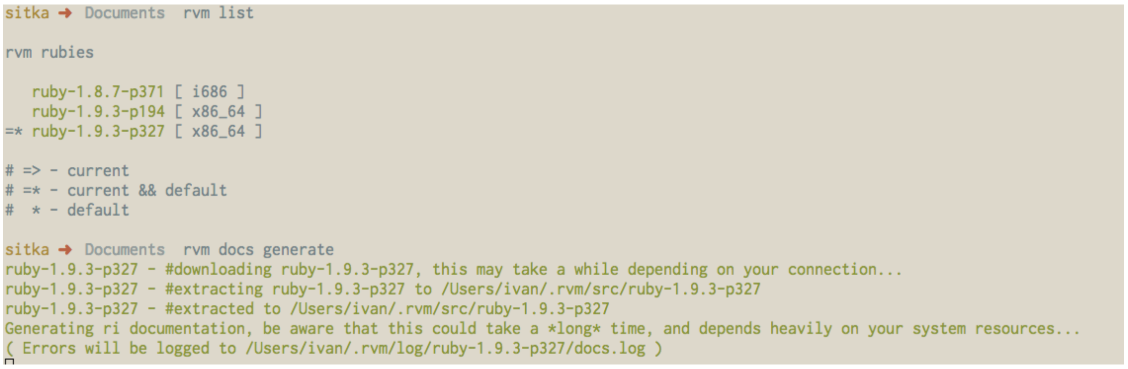
Pry

In this workshop we'll go over the pry REPL gem

Get your RI docs generated with RVM



```
sitka ➜ Documents  rvm list

rvm rubies

   ruby-1.8.7-p371 [ i686 ]
   ruby-1.9.3-p194 [ x86_64 ]
=* ruby-1.9.3-p327 [ x86_64 ]

# => - current
# =* - current && default
#  * - default

sitka ➜ Documents  rvm docs generate
ruby-1.9.3-p327 - #downloading ruby-1.9.3-p327, this may take a while depending on your connection...
ruby-1.9.3-p327 - #extracting ruby-1.9.3-p327 to /Users/ivan/.rvm/src/ruby-1.9.3-p327
ruby-1.9.3-p327 - #extracted to /Users/ivan/.rvm/src/ruby-1.9.3-p327
Generating ri documentation, be aware that this could take a *long* time, and depends heavily on your system resources...
( Errors will be logged to /Users/ivan/.rvm/log/ruby-1.9.3-p327/docs.log )
```

1. If you're on a laptop with a small SSD drive, and want to save space, only do this for one ruby.
Make sure you're in your most recent , or most commonly used, ruby by typing: rvm list

2. Then type: rvm docs generate
This will take quite a while, but it's totally worth it.

Watch the Pry Screencast

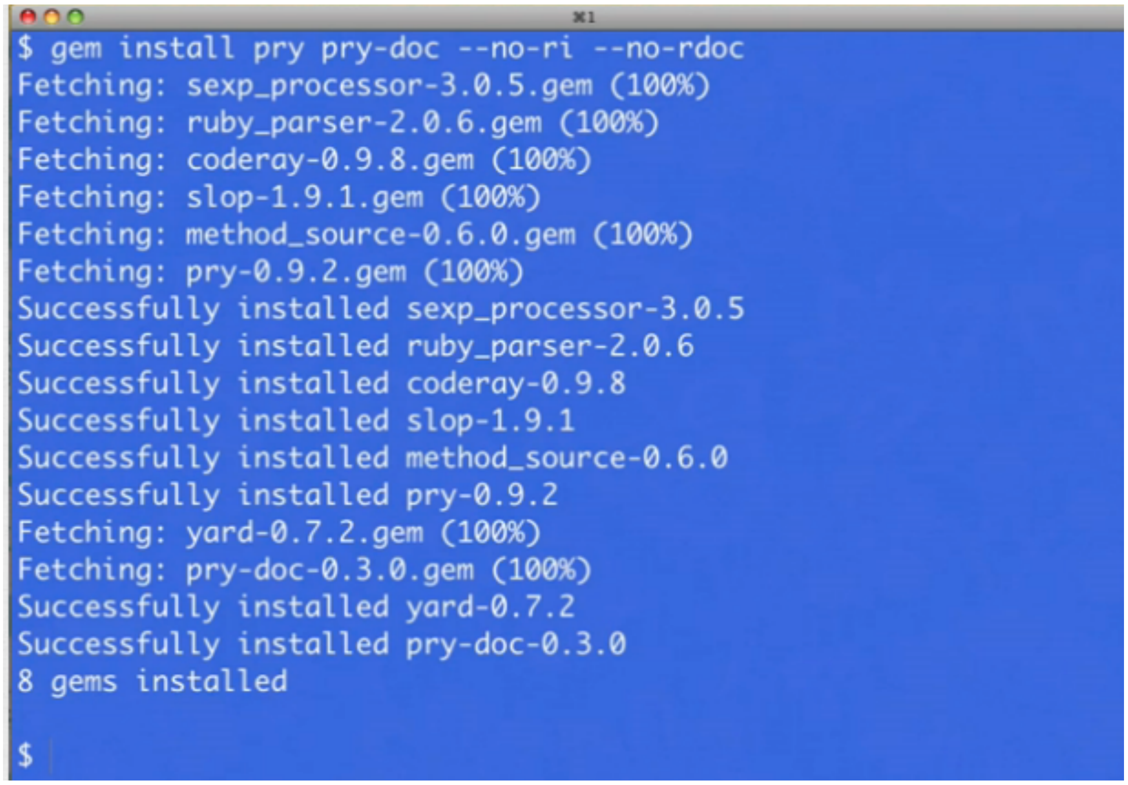
Introductory screencast

It is recommended you watch the following screencast put together by Josh Cheek. It includes instructions on installing Pry as well as some coverage of core features.

Watch other screencasts

Go to: http://pryrepl.org/screencasts.html
For Intro to ruby class, only up to 13m:48s
For Rails class, the whole thing

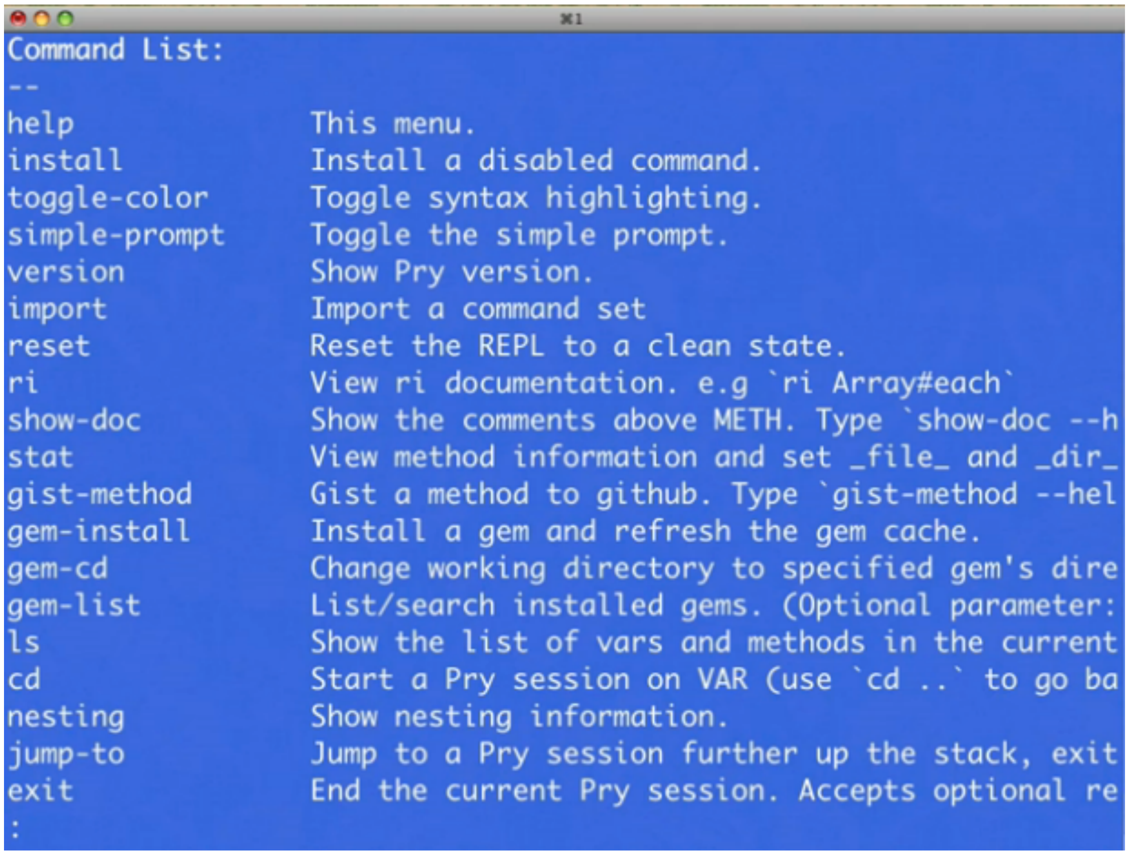
Install Pry



```
$ gem install pry pry-doc --no-ri --no-rdoc
Fetching: sexp_processor-3.0.5.gem (100%)
Fetching: ruby_parser-2.0.6.gem (100%)
Fetching: coderay-0.9.8.gem (100%)
Fetching: slop-1.9.1.gem (100%)
Fetching: method_source-0.6.0.gem (100%)
Fetching: pry-0.9.2.gem (100%)
Successfully installed sexp_processor-3.0.5
Successfully installed ruby_parser-2.0.6
Successfully installed coderay-0.9.8
Successfully installed slop-1.9.1
Successfully installed method_source-0.6.0
Successfully installed pry-0.9.2
Fetching: yard-0.7.2.gem (100%)
Fetching: pry-doc-0.3.0.gem (100%)
Successfully installed yard-0.7.2
Successfully installed pry-doc-0.3.0
8 gems installed

$
```

gem install pry pry-doc
personally, I like the docs... so skip doing --no-ri --no-rdoc unless you are really short on drive space

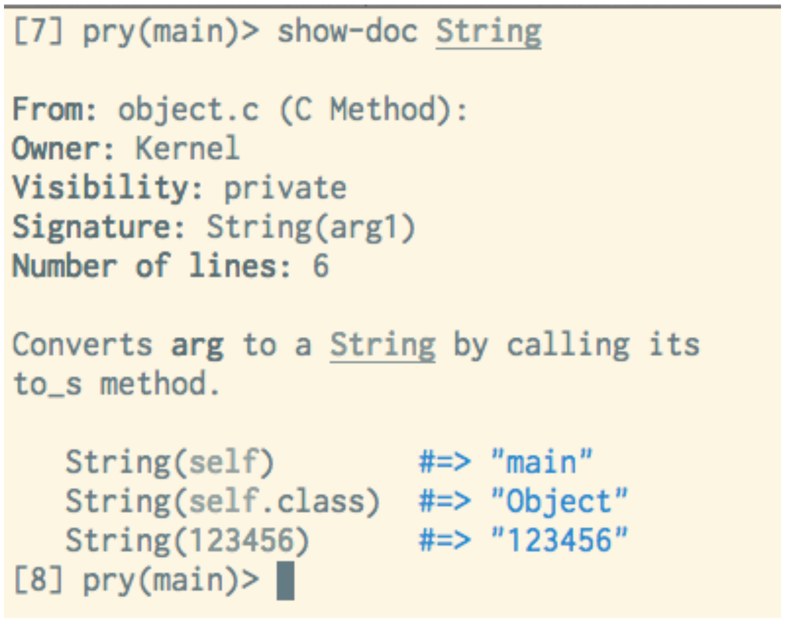
HALP! no, I mean help.



```
Command List:
--
help              This menu.
install           Install a disabled command.
toggle-color      Toggle syntax highlighting.
simple-prompt     Toggle the simple prompt.
version           Show Pry version.
import            Import a command set
reset             Reset the REPL to a clean state.
ri                View ri documentation. e.g `ri Array#each`
show-doc          Show the comments above METH. Type `show-doc --h
stat              View method information and set _file_ and _dir_
gist-method       Gist a method to github. Type `gist-method --hel
gem-install       Install a gem and refresh the gem cache.
gem-cd            Change working directory to specified gem's dire
gem-list          List/search installed gems. (Optional parameter:
ls                Show the list of vars and methods in the current
cd                Start a Pry session on VAR (use `cd ..` to go ba
nesting           Show nesting information.
jump-to           Jump to a Pry session further up the stack, exit
exit              End the current Pry session. Accepts optional re
:
```

Get an overview of the commands. Try getting help on any that interest you.

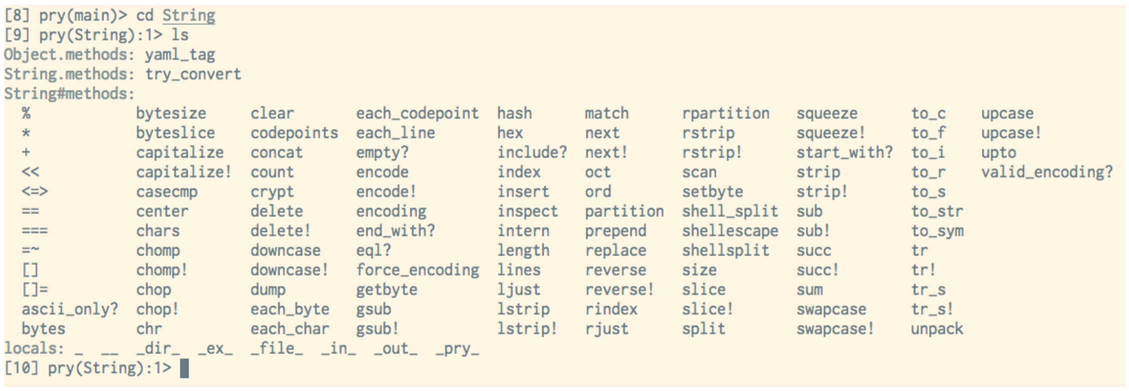
Let's check out the Documentation on String



```
[7] pry(main)> show-doc String

From: object.c (C Method):
Owner: Kernel
Visibility: private
Signature: String(arg1)
Number of lines: 6

Converts arg to a String by calling its
to_s method.

   String(self)        #=> "main"
   String(self.class)  #=> "Object"
   String(123456)      #=> "123456"
[8] pry(main)>
```

show-doc String

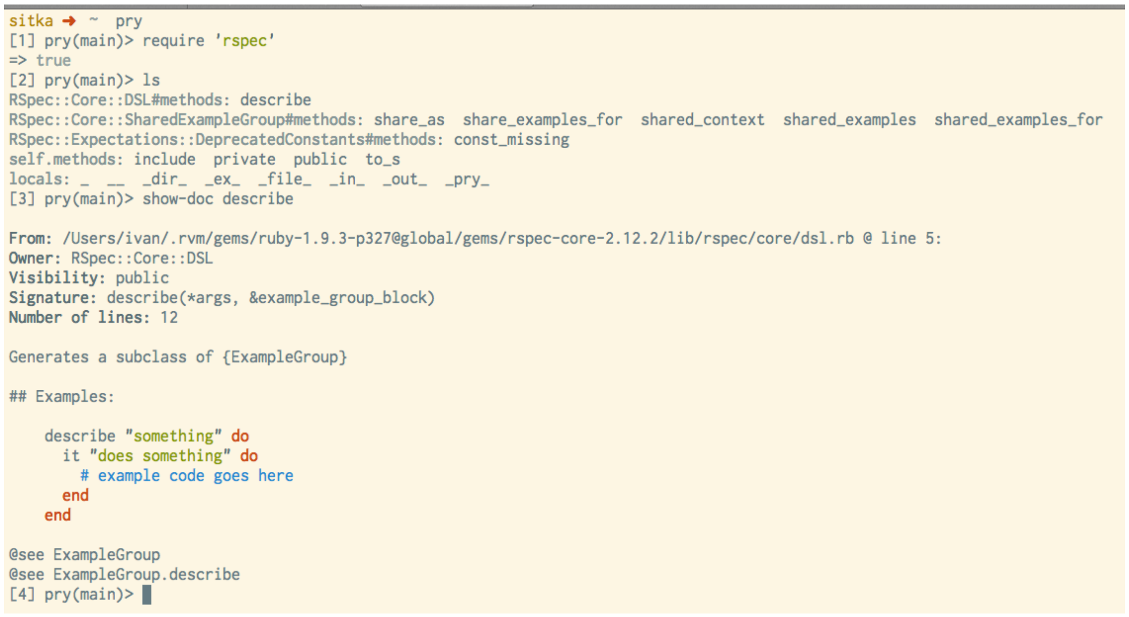
Let's cd and ls String. Yes, cd into a language primitive



```
[8] pry(main)> cd String
[9] pry(String):1> ls
Object.methods: yaml_tag
String.methods: try_convert
String#methods:
  %            bytesize     clear       each_codepoint  hash      match      rpartition   squeeze      to_c    upcase
  *            byteslice    codepoints  each_line       hex       next       rstrip       squeeze!     to_f    upcase!
  +            capitalize   concat      empty?          include?  next!      rstrip!      start_with?  to_i    upto
  <<           capitalize!  count       encode          index     oct        scan         strip        to_r    valid_encoding?
  <=>          casecmp      crypt       encode!         insert    ord        setbyte      strip!       to_s
  ==           center       delete      encoding        inspect   partition  shell_split  sub          to_str
  ===          chars        delete!     end_with?       intern    prepend    shellescape  sub!         to_sym
  =~           chomp        downcase    eql?            length    replace    shellsplit   succ         tr
  []           chomp!       downcase!   force_encoding  lines     reverse    size         succ!        tr!
  []=          chop         dump        getbyte         ljust     reverse!   slice        sum          tr_s
  ascii_only?  chop!        each_byte   gsub            lstrip    rindex     slice!       swapcase     tr_s!
  bytes        chr          each_char   gsub!           lstrip!   rjust      split        swapcase!    unpack
locals: _  __  _dir_  _ex_  _file_  _in_  _out_  _pry_
[10] pry(String):1>
```

cd String
ls
show-doc slice
whereami
Wow, look at all those useful methods!
TIP: The pager is most likely less by default. Press Q to exit the pager. you can common vim keys to navigate, roo.

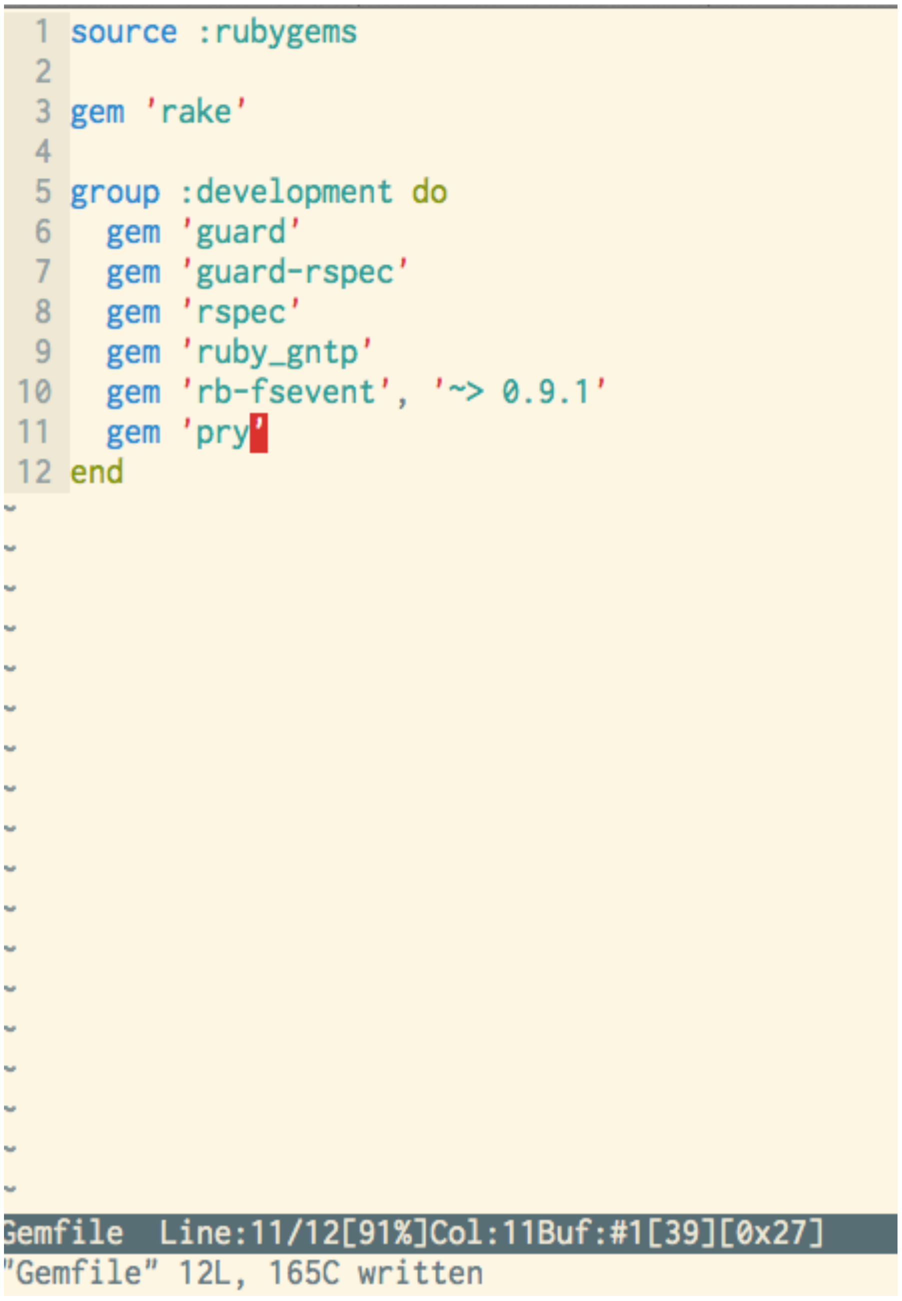
Let's look at rSpec



```
sitka ➜ ~  pry
[1] pry(main)> require 'rspec'
=> true
[2] pry(main)> ls
RSpec::Core::DSL#methods: describe
RSpec::Core::SharedExampleGroup#methods: share_as  share_examples_for  shared_context  shared_examples  shared_examples_for
RSpec::Expectations::DeprecatedConstants#methods: const_missing
self.methods: include  private  public  to_s
locals: _  __  _dir_  _ex_  _file_  _in_  _out_  _pry_
[3] pry(main)> show-doc describe

From: /Users/ivan/.rvm/gems/ruby-1.9.3-p327@global/gems/rspec-core-2.12.2/lib/rspec/core/dsl.rb @ line 5:
Owner: RSpec::Core::DSL
Visibility: public
Signature: describe(*args, &example_group_block)
Number of lines: 12

Generates a subclass of {ExampleGroup}

## Examples:

    describe "something" do
      it "does something" do
        # example code goes here
      end
    end

@see ExampleGroup
@see ExampleGroup.describe
[4] pry(main)>
```

cd / to get back to the top level if you're not there, or just re-start pry
require 'rspec'
ls
show-doc describe

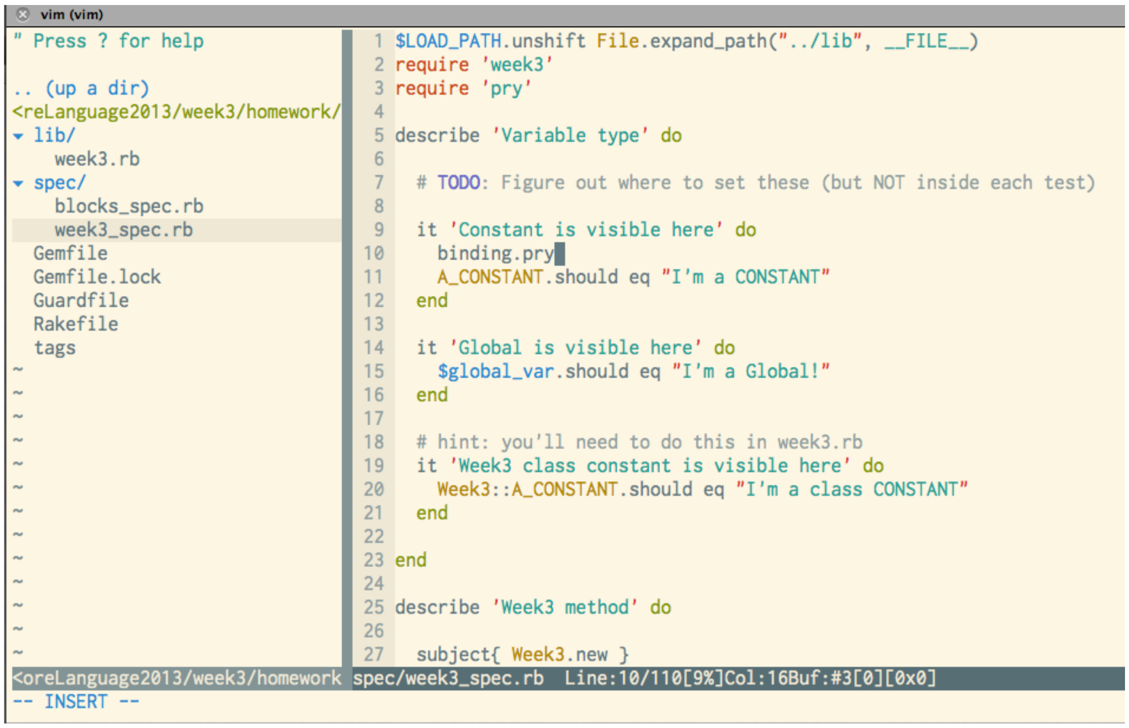
OK, let's add it to your week3 homework



```
source :rubygems

gem 'rake'

group :development do
  gem 'guard'
  gem 'guard-rspec'
  gem 'rspec'
  gem 'ruby_gntp'
  gem 'rb-fsevent', '~> 0.9.1'
  gem 'pry'
end
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
Gemfile  Line:11/12[91%]Col:11Buf:#1[39][0x27]
"Gemfile" 12L, 165C written
```

Add pry to your gemfile and bundle install

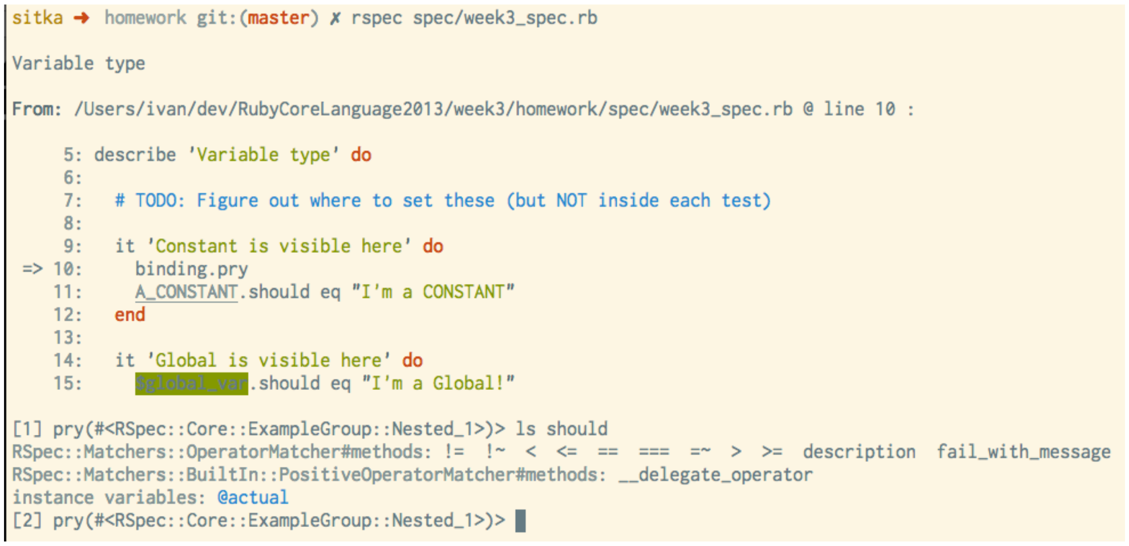
Add pry to your spec file



```
vim (vim)
" Press ? for help                            1 $LOAD_PATH.unshift File.expand_path("../lib", __FILE__)
                                              2 require 'week3'
.. (up a dir)                                 3 require 'pry'
<reLanguage2013/week3/homework/               4
▾ lib/                                        5 describe 'Variable type' do
    week3.rb                                  6
▾ spec/                                       7   # TODO: Figure out where to set these (but NOT inside each test)
    blocks_spec.rb                            8
    week3_spec.rb                             9   it 'Constant is visible here' do
  Gemfile                                    10     binding.pry
  Gemfile.lock                               11     A_CONSTANT.should eq "I'm a CONSTANT"
  Guardfile                                  12   end
  Rakefile                                   13
  tags                                       14   it 'Global is visible here' do
~                                            15     $global_var.should eq "I'm a Global!"
~                                            16   end
~                                            17
~                                            18   # hint: you'll need to do this in week3.rb
~                                            19   it 'Week3 class constant is visible here' do
~                                            20     Week3::A_CONSTANT.should eq "I'm a class CONSTANT"
~                                            21   end
~                                            22
~                                            23 end
~                                            24
~                                            25 describe 'Week3 method' do
~                                            26
~                                            27   subject{ Week3.new }
<oreLanguage2013/week3/homework spec/week3_spec.rb  Line:10/110[9%]Col:16Buf:#3[0][0x0]
-- INSERT --
```

note line 3 above and line 10. we want to break into pry in the first test

i'm in UR RSPEC!



```
sitka ➜  homework git:(master) ✗ rspec spec/week3_spec.rb

Variable type

From: /Users/ivan/dev/RubyCoreLanguage2013/week3/homework/spec/week3_spec.rb @ line 10 :

     5: describe 'Variable type' do
     6:
     7:   # TODO: Figure out where to set these (but NOT inside each test)
     8:
     9:   it 'Constant is visible here' do
 => 10:     binding.pry
    11:     A_CONSTANT.should eq "I'm a CONSTANT"
    12:   end
    13:
    14:   it 'Global is visible here' do
    15:     $global_var.should eq "I'm a Global!"

[1] pry(#<RSpec::Core::ExampleGroup::Nested_1>)> ls should
RSpec::Matchers::OperatorMatcher#methods: !=  !~  <  <=  ==  ===  =~  >  >=  description  fail_with_message
RSpec::Matchers::BuiltIn::PositiveOperatorMatcher#methods: __delegate_operator
instance variables: @actual
[2] pry(#<RSpec::Core::ExampleGroup::Nested_1>)>
```

Use Pry whenever this happens:

And the rest is up to you!