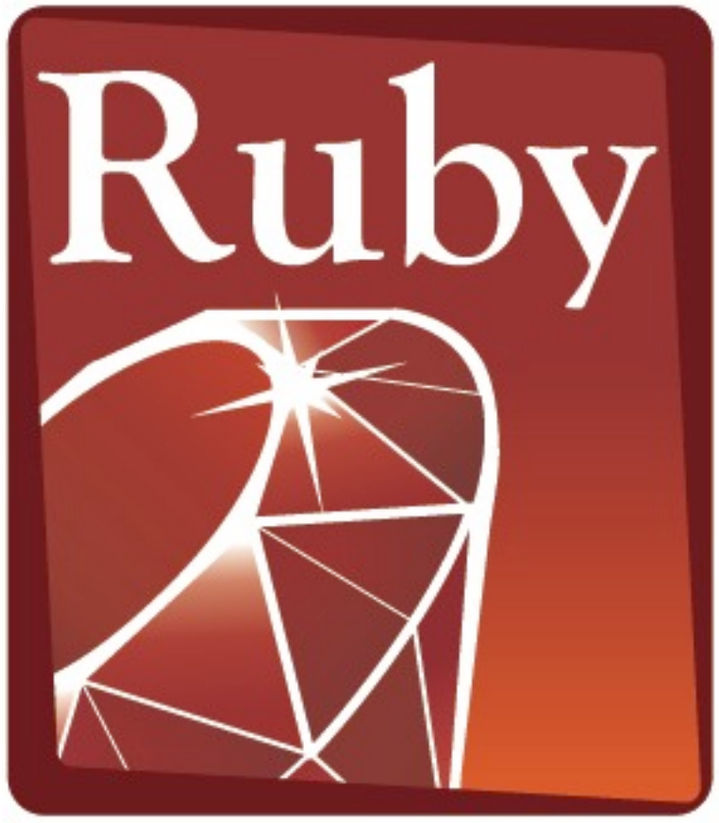

Ruby: The Core Language

Ruby Programming Certificate
UW PCE CP110
Winter 2013
Week 5

Stuff

- Week 1/2/3 homework due next week
- Pre-class Q&A again for week 6

Agenda

- Final Project
  - Ideas discussion & LocalwikiClient gem
- Guest Speaker:
  Nell Shamrell on Regular Expressions
- Week 4 Homework Review
- RubyGems
- Modules, Classes, File
- Week 5 Homework / Lab

Panel discussion:
Careers and Contributing to OSS

- During final class - 7:30pm on March 14
- AWESOME Guest Speakers!
- Submit topic ideas to me
- Please prepare questions in advance

Guest Speaker:
Nell Shamrell

Regular Expressions

Wk5 Homework

- Bad news: A hard week
- Good news: Largely in class.

Wk4 Homework Review

- Let's do it together!

Break!

Ruby Gems

- Packaged Ruby Code
- Available for Everyone!
- ruby-toolbox.com

Gem Structure

```
test_gem.gemspec
lib
  test_gem.rb
```

GemSpec

```
Gem::Specification.new do |s|
  s.name        = 'test_gem'
  s.version     = '0.0.1'
  s.date        = '2013-02-07'
  s.summary     = "Making a Test Gem"
  s.description = "A gem to explain how to make gems"
  s.authors     = ["Brandon Faloona"]
  s.email       = 'brandon@faloona.net'
  s.homepage    = 'http://rubygems.org/gems/test_gem'
  s.files       = ["lib/test_gem.rb"]
end
```

Building Your Gem

```
$ gem build test_gem.gemspec

 Successfully built RubyGem

 Name: test_gem

 Version: 0.0.1

 File: test_gem-0.0.1.gem
```

Installing Your Gem

```
$ gem install test_gem-0.0.1.gem

Successfully installed test_gem-0.0.1

1 gem installed

Installing ri documentation for
test_gem-0.0.1...

Installing RDoc documentation for
test_gem-0.0.1...
```

RubyGems.org

- Setup an Account:
  - https://rubygems.org/users/new
- Setup Credentials:
  - https://rubygems.org/profile/edit

Releasing Your Gem

```
$ gem push test_gem-0.0.1.gem
```

```
Pushing gem to RubyGems.org...

Successfully registered gem: test_gem
(0.0.1)
```

Viewing Remote Gems

```
$  gem list -r test_gem
```

```
*** REMOTE GEMS ***

test_gem (0.0.1)
```

Installing a Gem

```
$ gem install test_gem
Fetching: test_gem-0.0.1.gem (100%)
Successfully installed test_gem-0.0.1
1 gem installed
Installing ri documentation for
test_gem-0.0.1...
Installing RDoc documentation for
test_gem-0.0.1...
```

Adding CLI

- Command Line Interface
- Structure:

```
bin
    test_gem
```

- Include a SheBang

```
#!/usr/bin/env ruby
```

- .gemspec:

```
s.executables << 'test_gem'
```

Binary Arguments

- Use the ARGV Constant
  - e.g.: `ARGV[0]`
- An array of space delimited arguments

Test file structure

```
spec
  test_gem_spec.rb
RakeFile
.rspec
```

Rakefile

- Test task should be the default!

```
require 'rspec/core/rake_task'

RSpec::Core::RakeTask.new('spec')

task :default => :spec
```

.rspec

```
--color
--format documentation
```

Class variables

- @@count
- Shared across ALL instances of the class

Class Methods

```
class Dog

  @@population = 0

  def self.population
    @@population
  end

  def initialize
    @@population += 1
  end

end

Dog.new; Dog.new; Dog.population
=> 2
```

class << self

```
class Dog
  @@population = 0
  class << self
    def population
      @@population
    end
  end
end
```

Using Modules

```
class Contacts

  # mixin the Enumerable module
  include Enumerable

  def each
    # define how this class iterates
  end
end


# now Contacts has #sort, #grep, #map,
#   #partition, and much more.
```

Modules are like Classes

- Define a namespace

```
MyApp::Storage::File.open()
```

- Define CONSTANTS

```
MyApp::Storage::BASE_PATH
```

- Define class methods
- Define instance methods

Modules are NOT like Classes

- Can't be instantiated
  `Enumerable.new # => NoMethodError`
- Can't inherit from another Module

Defining Modules

```
module Storage

  def self.devices
    # creates Storage.devices
  end

  def write data
    # creates Storage#write
  end

end
```

Modules can share data with Classes

- @instance_variables and @@class_variables are shared with the classes that include them

Module example

```
module MyApp

  class File < ::File
    BASE_PATH = '/'
    # my
  end

end

MyApp::File::BASE_PATH
=> '/'
```

Method Visibility

- public
  - visible everywhere (default)
- private
  - visible internally
  - visible to child classes
- protected
  - rarely used
  - visible to objects with the correct superclass

Files

```
file = File.open(“test_file”, “r+”)
```

- "r" Read-only, starts at beginning of file (default)
- "r+" Read-write, starts at beginning of file
- "w" Write-only, truncates existing file to zero length or creates a new file for writing
- "w+" Read-write, truncates existing file to zero length or creates a new file for reading and writing

Files with Blocks

```
File.open(“test_file”, “r+”) do |f|
  f << “Add some text”
end
```

```
f = File.open(“roster.txt”)
f.each{ |line| puts line}
f.map{|line| line.split(‘,’).first}
```

Files

```
f = File.open(“roster.txt”)

f.gets (get a line)

f.pos (character position)

f.rewind (pos = 0)

f.read (file as a string)

f.readlines (array of each line as a
string)
```

Directories

```
Dir.pwd

Dir.chdir("..")

All files: Dir["*"]  All hidden files: Dir[".*"]

d = Dir.new(".")

Dir.mkdir("test")

d.entries

d.count

d.each{|f| puts f}
```

Wk 5 Homework