Ruby: The Core Language

Ruby Programming Certificate
UW PCE CP110
Winter 2013

Matz

wanted to make Ruby
“natural, not simple"

"Ruby is simple in appearance, but is very complex inside,
just like our human body."

Hal Fulton, The Ruby Way, 2001
http://www.infoq.com/articles/what-is-the-ruby-way

Community

- Websites galore
  - Tutorials
  - Reference
  - Subscriptions

Try Ruby

- IRB in your browser!
- http://tryruby.com/

Ruby Koans

- Step wise, test driven tutorial
- http://rubykoans.com/

Ruby Doc

- The official ruby documentation site
  - http://www.ruby-doc.org/
- Use API Dock as an alternate if you like
  - http://apidock.com/ruby/
  - http://apidock.com/rspec/

Ruby QuickRef

- Ryan Davis is nice, smart, opinionated, and prolific.
- He wrote this ages ago and keeps it up to date.
- (nearly) Everything on one page!
- http://zenspider.com/Languages/Ruby/QuickRef.html

Subscriptions

- Ruby Rogues http://rubyrogues.com/
- RubyMonk.com http://rubymonk.com/
- RubyTapas - http://devblog.avdi.org/rubytapas/
- MetaCasts.tv - http://www.metacasts.tv/

Check In

- Canvas
  - Spam
  - Discussions
  - Photos
- Feedback
  - During break, in e-mail

Final Project

- ClassRoster in /final_project/class_roster
- LocalwikiClient

LocalwikiClient

https://github.com/bfaloona/localwiki_client

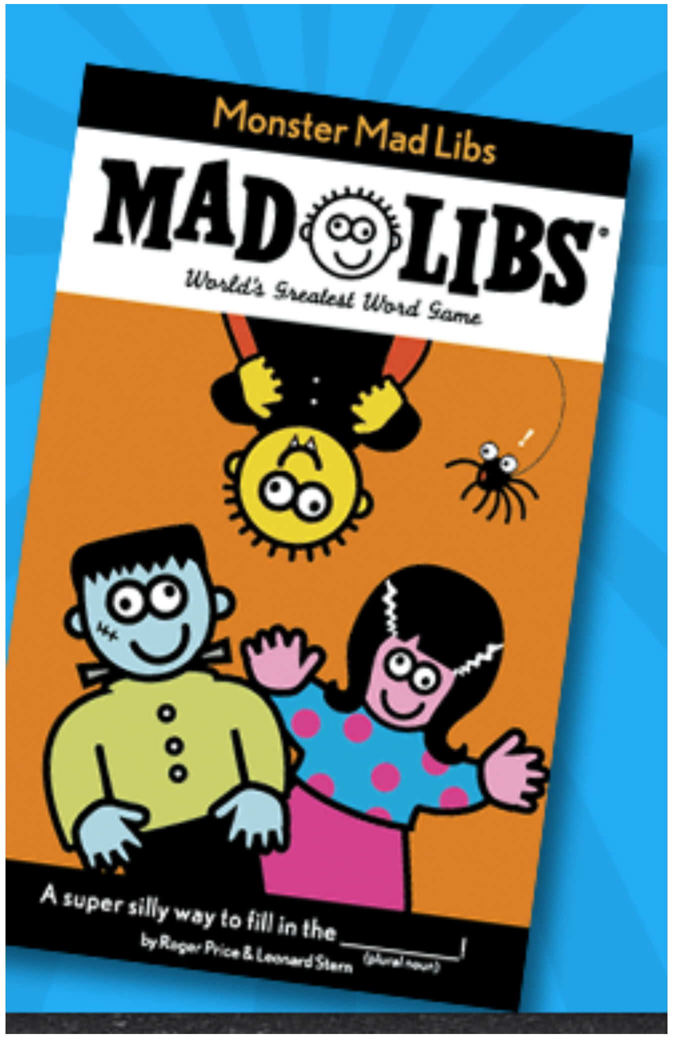
- Structure - lib, spec, README.md, History.txt
- Gemfile, .gemspec, Rakefile
- Feature Branches
- Travis CI
- Flog rake task or Code Climate
- Wiki, Issues

In Class Exercises

- Mad Libs
- Group Exercise!

Mad Libs

```
/week2/exercises/mad_libs.rb
ruby mad_libs.rb
```


Monster Mad Libs
MAD LIBS
World's Greatest Word Game
A super silly way to fill in the ____
(plural noun)
by Roger Price & Leonard Stern

- noun = gets.chomp
- adjective = gets.chomp
- verb_past_tense = gets.chomp
- “The #{adjective} #{noun} #{verb_past_tense} past the graveyard”

Group exercise!

- Build a gem with 2 or three other classmates!
- Stand up if you (kinda) understand git/github
- Split into 6 groups. GO!!!
- Pick ONE laptop. Make room, gather round, Say hi.
- Build a (toy) Gem! e.g. Dog.new.say => “Woof!”

Make a GEM!

- Write README.md
- Create empty git + github repository
- Build file structure. Gemfile, .gemspec file
- Write a spec, run it, watch it fail!
- Commit. Push.
- Each person clone to their laptop.
- Make the spec pass. Commit the change.
- Build the gem! (`$ gem build`)
- Install the gem (`$ gem install mygem.gem --local`)
- Write a script that uses it! (`$ require ‘mygem’`)

Lightning Talk!

or

Present your Homework!

Homework Review

Next week’s homework

- Specs are written
- You write the code
- /week2
    - README.md
    - /homework
- Let’s fix one spec!

Array

- List of things. Untyped.
- Mutable. Ordered.
- Literal syntax (although Array.new can be useful)
- Access with zero based index or negative index from end
- Tons of operators and methods
    - + << | * .push .pop .sort .uniq
- The power of Enumerable

Array

```
["1", "2", "3"]
Array.new ["1", "2", "3"]
%w[1 2 3]

a = []
a << "1"
a << "2"
a << "3"
["1", "2", "3"]
a[0] => “1”
```

Hash

- Key / Value pairs
- Symbols make VERY efficient keys
- value is untyped
- Ordered (as of Ruby 1.9)
- You guessed it, Enumerable!

Hash

```
# ruby 1.8 & 1.9
{ :a => 1, :b => "2", :c => 3.0 }

# exact same, but ruby 1.9 only
{ a: 1, b: “2”, c: 3.0 }

my_hash = {}
my_hash[:a] = 1
my_hash[:b] = "2"
my_hash[:c] = 3.0
```

Iterators

- The magic of blocks! (don’t worry we’ll come back to blocks)
- 4.times { puts ‘Iterators repeat\n” }
- my_array.each { |item| puts item }
- my_array.collect { |item| item * 2 }

Homework!

- Reading assignment
- Homework
  Instructions online:
- Quiz