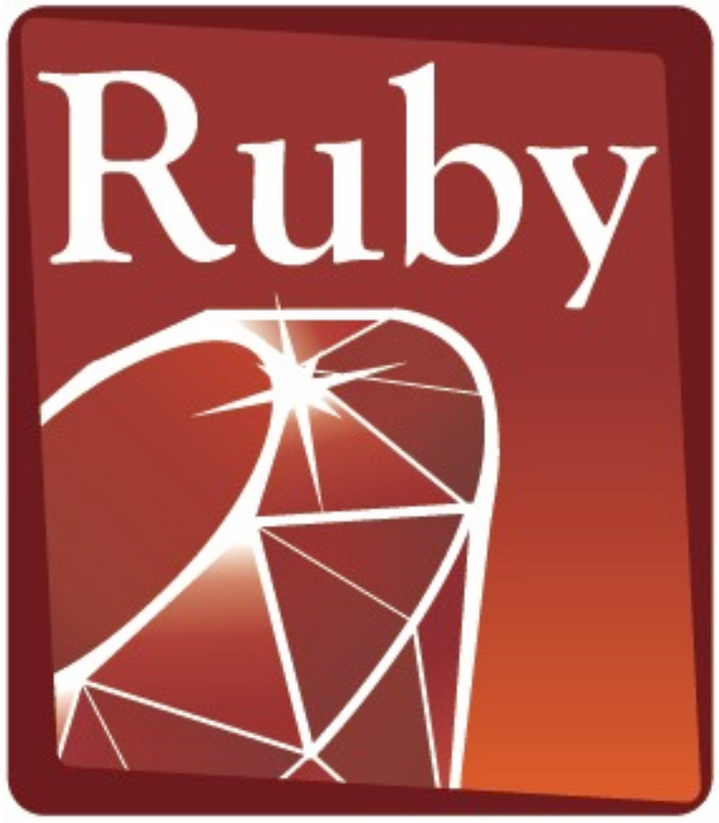

Ruby: The Core Language

Ruby Programming Certificate
UW PCE CP110
Winter 2013
Week 6

Empowerment of individuals is a key part of what makes open source work, since in the end, innovations tend to come from small groups, not from large, structured efforts.

-Tim O'Reilly

Stuff

- Need to end a bit early tonight (~9pm)
- Next week: Guest Speaker on Rake
- Collaborate/Help eachother:
  - Seattle.rb this last Tuesday?
  - Before class next Thursday

Agenda

- Final Project
  - Requirements
  - Feature Lists, Pending Specs
  - Lab
- Week 5 Homework Review
- Exceptions and Exception Handling
- Test Doubles
- Week 6 Homework / Lab
- LocalwikiClient group project

Community:

Contributing to Open Source Software

- Why?
  - Free Speech vs. Free Beer
    - You have a blocking bug
    - Fixed... But later you want to upgrade to latest version
- How?
  - Video: Aaron Patterson

Wk6 Homework

- Extend last week's test_gem
  - Exception handling
  - Config files
  - Testing with Stubs and Mocks

Wk5 Homework Review

- Let's do it together!

Final Project

- Requirements
- Feature List
- Pending specs
- Lab

Break!

Testing with Stubs and Mocks

- Test the right thing
- Make your tests fast
- Stubs are dumb
- Mocks create expectations
- Also called: Doubles, Fakes, Stubs, Mocks, Shims

Detour!
HTTP client: Faraday

```
$ gem install faraday

$ irb
> require 'faraday'
=> true
> resp = Faraday.get 'http://www.google.com/'
> resp.status
=> 200
> resp.body
=> "<!doctype html><html itemtype=\" ... "
```

How do I test this method off the network?

```
def status url
  Faraday.get(url).status
end
```

Exceptions

- Classes that support raise alarm
- Instance methods
  - #message
  - #backtrace

Rescuing an Exception

```
def my_method
  begin

    do_something_dangerous!

  rescue StandardError => e
    # handle the error
    do_something_less_dangerous

  ensure
    Logger.log('my_method completed')
    Logger.log(e.message) if e

  end

end
```

Implicit begin block

```
def my_method

  do_something_dangerous!

rescue StandardError => e
  # handle the error
  do_something_less_dangerous

ensure
  Logger.log('my_method completed')
  Logger.log(e.message) if e

end
```

Raising an Exception

```
def my_method
  result = do_something_dangerous
  if result.nil?
    raise RuntimeError, 'We have a problem'
  end
end
```

Defining an Exception

```
class MySpecialException < StandardError

end
```

Exception Hierarchy

```
Exception
  NoMemoryError
  ScriptError
    LoadError
    NotImplementedError
    SyntaxError
SignalException
  Interrupt
    Timeout::Error
StandardError          # caught by rescue
  ...
```

```
StandardError           # caught by rescue
  ArgumentError
  IOError
    EOFError
  IndexError
  LocalJumpError
  NameError
    NoMethodError
  RangeError
    FloatDomainError
  RegexpError
  RuntimeError
    Timeout::Error      # >= ruby 1.9.2
  SecurityError
  SocketError
  SystemCallError
  SystemStackError
  ThreadError
  TypeError
  ZeroDivisionError
SystemExit
fatal
```

Lab

- Week 6 Homework
- LocalwikiClient group project