Under the Hood

An intimate tour of some gems among gems

David Chelimsky
Articulated Man, Chicago, IL, USA

What are Ruby Gems?

- **RubyGems** is a package management framework for the Ruby language
  - Similar idea to:
    - maven (Java)
    - apt-get (debian linux)
    - emerge (gentoo linux)
- **Gems** are the packaged applications and libraries themselves

What are Ruby Gems?

- **RubyGems** supports:
  - Creation/publishing of gems
  - "Rational" Versioning Policy
  - Access to published gems
    - Transparent installation including:
      - ruby libraries
      - shell commands/scripts
- Dependency management

What are Ruby Gems?

- **Gems** can be
  - component libraries
  - transparent plugins
  - complete applications
  - shell commands/scripts
  - any combination of the above

RubyForge Stats

- As of 13 March, 2008
  - 5,409 Hosted Projects
  - Millions of gem downloads
  - Millions of source checkouts

Installation

- http://www.rubygems.org/read/chapter/3

```
$ wget http://rubyforge.org/frs/download.php/29548/rubygems-1.0.1.tgz
$ tar -zxvf rubygems-1.0.1.tgz
$ cd rubygems-1.0.1
$ ruby setup.rb
```

Discovery

Help

```
$ gem help

  RubyGems is a sophisticated package manager for Ruby.  This is a
  basic help message containing pointers to more information.

    Usage:
      gem -h/--help
      gem -v/--version
      gem command [arguments...] [options...]

    Examples:
      gem install rake
      gem list --local
      gem build package.gemspec
      gem help install
...
```

Help (cont'd)

```
...
    Further help:
      gem help commands            list all 'gem' commands
      gem help examples            show some examples of usage
      gem help platforms           show information about platforms
      gem help <COMMAND>           show help on COMMAND
                                     (e.g. 'gem help install')
    Further information:
      http://rubygems.rubyforge.org
```

Environment

```
$ gem environment
RubyGems Environment:
  - RUBYGEMS VERSION: 1.0.1 (1.0.1)
  - RUBY VERSION: 1.8.6 (2007-09-24 patchlevel 111) [universal-darwin9.0]
  - INSTALLATION DIRECTORY: /Library/Ruby/Gems/1.8
  - RUBY EXECUTABLE: /System/Library/Frameworks/Ruby.framework/Versions/1.8/usr/bin/ruby
  - RUBYGEMS PLATFORMS:
    - ruby
    - universal-darwin-9
  - GEM PATHS:
    - /System/Library/Frameworks/Ruby.framework/Versions/1.8/usr/lib/ruby/gems/1.8
    - /Library/Ruby/Gems/1.8
  - GEM CONFIGURATION:
    - :update_sources => true
    - :verbose => true
    - :benchmark => false
    - :backtrace => false
    - :bulk_threshold => 1000
  - REMOTE SOURCES:
    - http://gems.rubyforge.org
```

Installed Gems

```
$ gem query --local

*** LOCAL GEMS ***

abstract (1.0.0)
actionmailer (2.0.2, 2.0.1, 1.3.6, 1.3.3)
actionpack (2.0.2, 2.0.1, 1.13.6, 1.13.3)
actionwebservice (1.2.6, 1.2.3)
activerecord (2.0.2, 2.0.1, 1.15.6, 1.15.3)
activeresource (2.0.2, 2.0.1)
activesupport (2.0.2, 2.0.1, 1.4.4, 1.4.2)
acts_as_ferret (0.4.1)
bones (1.3.1)
capistrano (2.1.0, 2.0.0)
cgi_multipart_eof_fix (2.5.0, 2.2)
coderay (0.7.4.215)
daemons (1.0.9, 1.0.7)
...
```

Available Gems

```
$ gem query --remote --details

*** REMOTE GEMS ***

aalib-ruby (0.7.1)
    a graphics context and input library for text terminals

abstract (1.0.0)
    a library which enable you to define abstract method in Ruby

...

zsff (1.0.0)
    A parser/validator for ZSFF feeds

zyps (0.7.6, 0.7.5, 0.7.4, 0.7.3, 0.7.2, 0.7.1, 0.7.0, 0.6.3, 0.6.2, 0.6.1, 0.5.2,
0.5.1, 0.4.1, 0.3.1, 0.2.1, 0.1.1)
    A game library for Ruby
```

Let's Get One!

Installing a Gem

```
$ gem install hpricot
Building native extensions.
Successfully installed hpricot-0.6
1 gem installed
Installing ri documentation for hpricot-0.6...
Installing RDoc documentation for hpricot-0.6...
```

ri

ri

```
$ ri Builder::XmlMarkup
------------------------------------ Class: Builder::XmlMarkup < XmlBase
     Create XML markup easily. All (well, almost all) methods sent to
     an XmlMarkup object will be translated to the equivalent XML
     markup. Any method with a block will be treated as an XML markup
     tag with nested markup in the block.

     Examples will demonstrate this easier than words. In the
     following, xm is an XmlMarkup object.

       xm.em("emphasized")             # => <em>emphasized</em>
       xm.em { xmm.b("emp & bold") }   # => <b><em>emph &amp; bold</em></b>
     ...
```

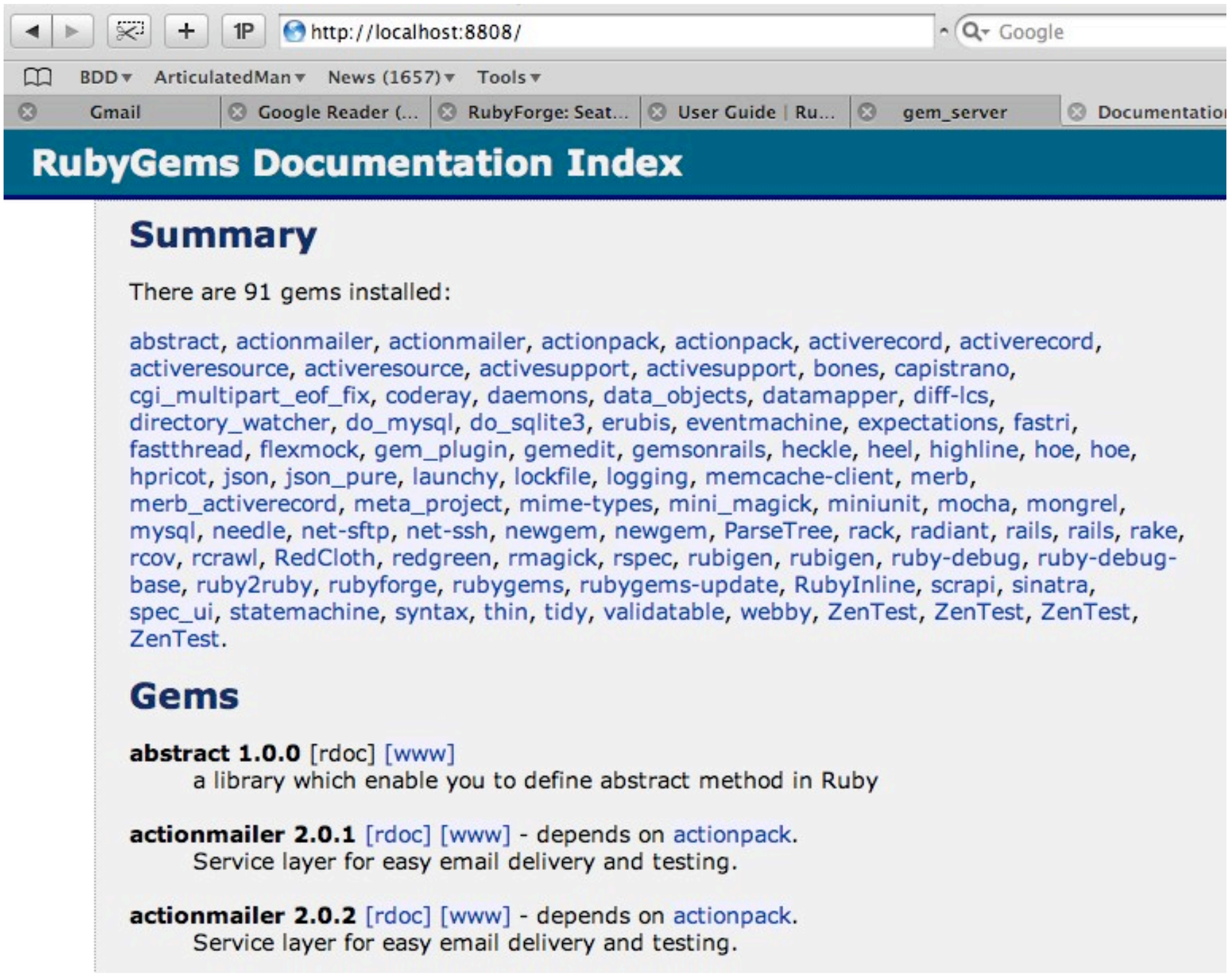
Gem Server

gem server

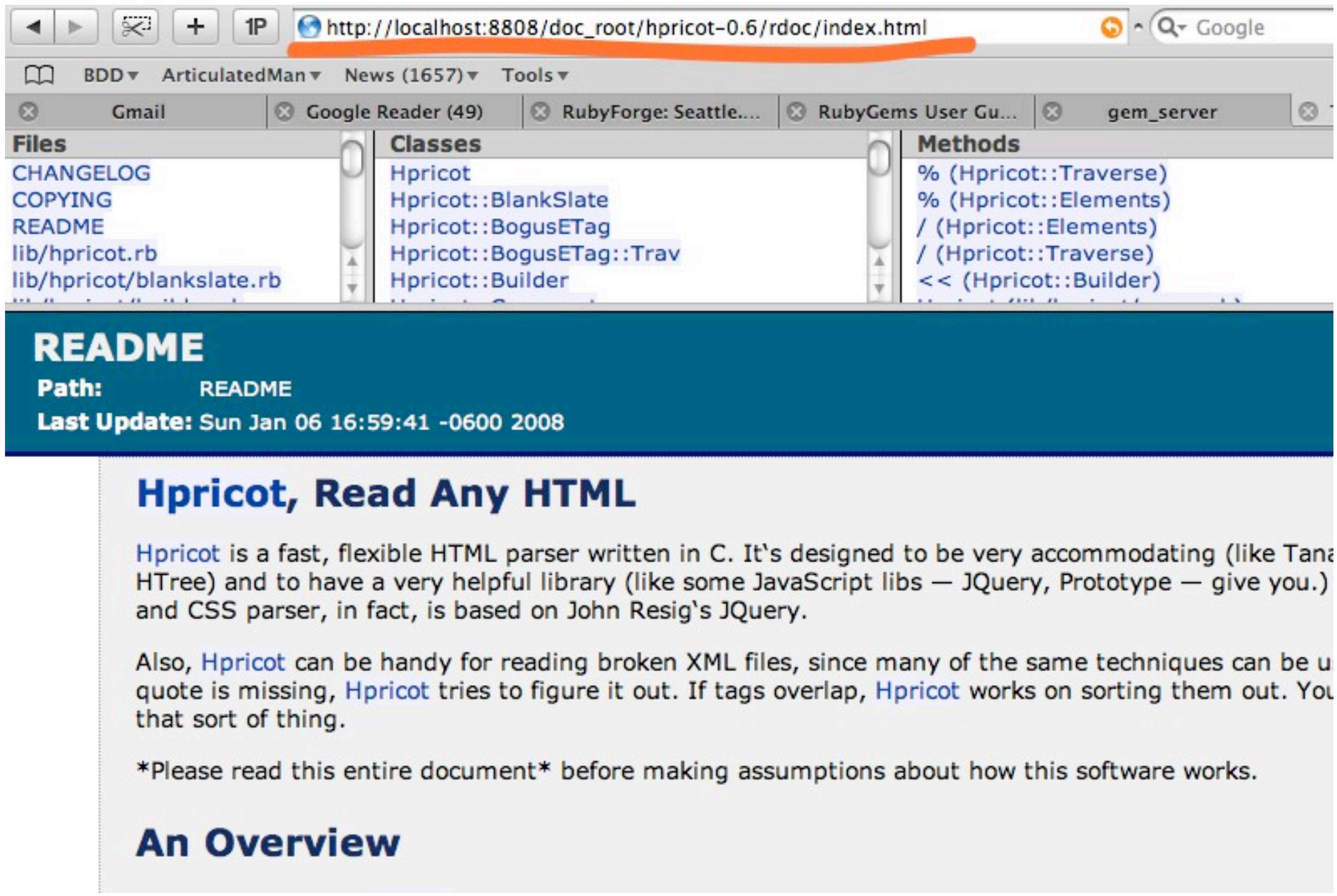
```
$ gem server
Starting gem server on http://localhost:8808/
```

RDoc

gem server


http://localhost:8808/
Google
BDD
ArticulatedMan
News (1657)
Tools
Gmail
Google Reader (...
RubyForge: Seat...
User Guide | Ru...
gem_server
RubyGems Documentation Index
Summary
There are 91 gems installed:
abstract, actionmailer, actionmailer, actionpack, actionpack, activerecord, activerecord, activeresource, activeresource, activesupport, activesupport, bones, capistrano, cgi_multipart_eof_fix, coderay, daemons, data_objects, datamapper, diff-lcs, directory_watcher, do_mysql, do_sqlite3, erubis, eventmachine, expectations, fastri, fastthread, flexmock, gem_plugin, gemedit, gemsonrails, heckle, heel, highline, hoe, hoe, hpricot, json, json_pure, launchy, lockfile, logging, memcache-client, merb, merb_activerecord, meta_project, mime-types, mini_magick, miniunit, mocha, mongrel, mysql, needle, net-sftp, net-ssh, newgem, newgem, ParseTree, rack, radiant, rails, rails, rake, rcov, rcrawl, RedCloth, redgreen, rmagick, rspec, rubigen, rubigen, ruby-debug, ruby-debug-base, ruby2ruby, rubyforge, rubygems, rubygems-update, RubyInline, scrapi, sinatra, spec_ui, statemachine, syntax, thin, tidy, validatable, webby, ZenTest, ZenTest, ZenTest, ZenTest.
Gems
abstract 1.0.0 [rdoc] [www]
a library which enable you to define abstract method in Ruby
actionmailer 2.0.1 [rdoc] [www] - depends on actionpack.
Service layer for easy email delivery and testing.
actionmailer 2.0.2 [rdoc] [www] - depends on actionpack.
Service layer for easy email delivery and testing.

gem server


http://localhost:8808/doc_root/hpricot-0.6/rdoc/index.html
Google
BDD
ArticulatedMan
News (1657)
Tools
Gmail
Google Reader (49)
RubyForge: Seattle....
RubyGems User Gu...
gem_server
Files
CHANGELOG
COPYING
README
lib/hpricot.rb
lib/hpricot/blankslate.rb
Classes
Hpricot
Hpricot::BlankSlate
Hpricot::BogusETag
Hpricot::BogusETag::Trav
Hpricot::Builder
Methods
% (Hpricot::Traverse)
% (Hpricot::Elements)
/ (Hpricot::Elements)
/ (Hpricot::Traverse)
<< (Hpricot::Builder)
README
Path: README
Last Update: Sun Jan 06 16:59:41 -0600 2008
Hpricot, Read Any HTML
Hpricot is a fast, flexible HTML parser written in C. It's designed to be very accommodating (like Tana
HTree) and to have a very helpful library (like some JavaScript libs — JQuery, Prototype — give you.)
and CSS parser, in fact, is based on John Resig's JQuery.
Also, Hpricot can be handy for reading broken XML files, since many of the same techniques can be u
quote is missing, Hpricot tries to figure it out. If tags overlap, Hpricot works on sorting them out. You
that sort of thing.
Please read this entire document before making assumptions about how this software works.
An Overview

Make Sure it Works!

Run the Tests with the gem command

```
$ gem check -t ZenTest -v 3.9.1
```

- Lets you know when there are
  - no tests
  - test failures
- Very few gems use this, so we need some other strategies ...

Unpack and Run the Specs with Rake

```
$ gem unpack statemachine
Unpacked gem: '/Users/david/statemachine-0.4.1'
$ cd statemachine-0.4.1/
$ rake spec
(in /Users/david/statemachine-0.4.1)
.....................................................
.........................

Finished in 0.061194 seconds

76 examples, 0 failures
```

Unpack and Run the Tests with Rake

```
$ gem unpack mocha
Unpacked gem: '/Users/david/mocha-0.5.6'
$ cd mocha-0.5.6/
$ rake test
(in /Users/david/mocha-0.5.6)
.....................................................
.....................................................
..............
Finished in 0.100842 seconds.

108 tests, 88 assertions, 0 failures, 0 errors
```

When all else fails ...

```
$ gem unpack hpricot
Unpacked gem: '/Users/david/hpricot-0.6'
$ cd hpricot-0.6/test
$ ruby -e "Dir['./test_*'].each {|f| require f}"
Loaded suite -e
Started
.....................................................
.......
Finished in 2.545982 seconds.

62 tests, 6145 assertions, 0 failures, 0 errors
```

hpricot

why the lucky stiff

hpricot

- A Fast, Enjoyable HTML Parser for Ruby
  - http://code.whytheluckystiff.net/hpricot/
- Uses Standard CSS Selectors
- Simple, very expressive syntax

hpricot example

```
require 'open-uri' # So we can open an URL
require 'rubygems'# It's not enabled by default
require 'hpricot'  # Looks in your installed gems

doc = Hpricot(open("http://www.infoq.com/"))
(doc / "div.entry h1 a").each do |a|
  puts a.inner_text
end
```

```
$ ruby hpricot_infoq.rb
Cockburn on Testing: Real Programmers have GUTs
Presentation: Designing for Testability
Riding the Waves of a new Ruby Web Framework
Pragmatic is the new black - Reality Driven Development
```

Lessons in Usability

hpricot

hpricot | usability

- easy access

```
require 'hpricot'
doc = Hpricot(open("http://www.infoq.com/"))
```

- supported by global method

```
def Hpricot(input = nil, opts = {}, &blk)
  Hpricot.parse(input, opts, &blk)
end
```

- this is just a shortcut
- the real method is standard OO fare:
  - object => message

hpricot | usability

- friendly syntax

```
(doc / "div.entry h1 a")
# => Hpricot::Elements < Array

(doc % "div.entry h1 a")
# => Hpricot::Elem (a single element)
```

- enabled by Ruby's treatment of operators

```
(doc / "div.entry h1 a") == (doc./("div.entry h1 a"))
```

- and a simple alias_method

```
alias_method :/, :search
```

mocha

James Mead

mocha

- A library for mocking and stubbing
  - http://mocha.rubyforge.org/
- Simple, intuitive DSL
- Supports mock objects with method stubbing and message expectations
- Adds method stubbing and message expectations to *real* objects as well.

mocha example

```ruby
messenger = mock()
messenger.expects(:<<).with("this message")
reporter = Reporter.new(messenger)
reporter.report("this message")
```

```
$ ruby mocha_example.rb
F

1)
Mocha::ExpectationError in 'Reporter should pass messages to messenger'
#<Mock:0x13b1004>.<<('this message') - expected calls: 1, actual calls: 0
mocha_example.rb:18:
/Library/Ruby/Site/1.8/spec.rb:20:in `run'
mocha_example.rb:15:

Finished in 0.06346 seconds

1 example, 1 failure
```

Lessons in Usability

mocha

mocha | usability

- easy access

```
require 'mocha'
include Mocha::AutoVerify
messenger = mock()
messenger.expects(:<<)
messenger.verify
# => examples/mocha_usability.rb:4: \
# => #<Mock:0x586a4c>.<<(any_parameters) \
# => - expected calls: 1, actual calls: 0 (Mocha::ExpectationError)
```

mocha | usability

- friendly DSL

```
mock.expects(:message).with(:arg).once
mock.expects(:message).with(:arg).times(2)
mock.expects(:message).returns(5)

mock.stubs(:name).returns("David")
```

mocha | usability

- automatic plugin for Test::Unit::TestCase

```
# in mocha.rb

module Test
  module Unit
    class TestCase
      include Mocha::Standalone
      include Mocha::TestCaseAdapter
    end
  end
end
```

- enabled by Ruby's Open Class Model

mocha | usability

- easily plugged into any other framework

```
# RSpec
# in plugins/mock_frameworks/mocha.rb
module Spec
  module Plugins
    module MockFramework
      include Mocha::Standalone
      def setup_mocks_for_rspec # used internally by RSpec
        mocha_setup             # from Mocha::Standalone
      end
      ...
    end
  end
end
```

builder

Jim Weirich

builder

- Provides a simple way to create XML markup and data structures.
- Simple, intuitive DSL

Lessons in Usability

builder

builder | usability

```ruby
require 'builder'

xml = Builder::XmlMarkup.new(:indent => 2)

language = xml.language do |language|
  language.name("Ruby")
  language.type("Dynamic")
end

puts language
```

```
$ ruby builder_example.rb
<language>
  <name>Ruby</name>
  <type>Dynamic</type>
</language>
$
```

builder | usability

```
# Create XML markup based on the name of the method.  This method
# is never invoked directly, but is called for each markup method
# in the markup block.
def method_missing(sym, *args, &block)
  text = nil
  attrs = nil
  sym = "#{sym}:#{args.shift}" if args.first.kind_of?(Symbol)
  ...
```

Creating Your Own Gems

Creating a Gem

```
require 'rake/gempackagetask'

spec = Gem::Specification.new do |s|
  s.name = 'object_identifier'
  s.version = '0.0.1'
  s.summary = 'Adds object_id to #inspect'
  s.files = FileList['lib/**/*','spec/**/
*','Rakefile']
end

Rake::GemPackageTask.new(spec) do |pkg|
end
```

```
$ rake package
  Successfully built RubyGem
  Name: object_identifier
  Version: 0.0.1
  File: object_identifier-0.0.1.gem
$ gem install object_identifier-0.0.1.gem
```

Gem Creation Tools

- sow (ships with hoe)
- newgem
- bones
- gemify

- All of these are gems themselves!

hoe

hoe

- A simple rake/rubygems helper for project Rakefiles
- Includes many useful tasks
- Viral: includes itself as a dependency in created gems
  - whether hoe is useful to the consumer of your gem or not

hoe: bootstrap a gem

```
$ sow object_identifier
creating project object_identifier
... done, now go fix all occurrences of 'FIX'

  ObjectIdentifier/Rakefile:9:  # p.author = 'FIX'
  ObjectIdentifier/Rakefile:10:  # p.email = 'FIX'
  ObjectIdentifier/Rakefile:11:  # p.summary = 'FIX'
  ObjectIdentifier/README.txt:3:* FIX (url)
  ObjectIdentifier/README.txt:7:FIX (describe your package)
  ObjectIdentifier/README.txt:11:* FIX (list of features or problems)
  ObjectIdentifier/README.txt:15:  FIX (code sample of usage)
  ObjectIdentifier/README.txt:19:* FIX (list of requirements)
  ObjectIdentifier/README.txt:23:* FIX (sudo gem install, anything else)
  ObjectIdentifier/README.txt:29:Copyright (c) 2008 FIX
```

hoe: installed rake tasks

```
$ cd ObjectIdentifier
$ rake -T
(in /Users/david/Documents/QCon/under_the_hood/gems/ObjectIdentifier)
Hoe email value not set - Fix by 2008-04-01!
Hoe author value not set - Fix by 2008-04-01!
rake announce          # Generate email announcement file and post to rubyfo...
rake audit             # Run ZenTest against the package
rake check_manifest    # Verify the manifest
rake clean             # Clean up all the extras
rake clobber_docs      # Remove rdoc products
rake clobber_package   # Remove package products
rake config_hoe        # Create a fresh ~/.hoerc file
rake debug_gem         # Show information about the gem.
rake default           # Run the default tasks
rake docs              # Build the docs HTML Files
rake email             # Generate email announcement file.
...
```

hoe: installed rake tasks

```
...
rake gem                # Build the gem file ObjectIdentifier-1.0.0.gem
rake generate_key       # Generate a key for signing your gems.
rake install            # Install the package.
rake install_gem        # Install the package as a ge
rake multi              # Run the test suite using multiruby
rake package            # Build all the packages
rake post_blog          # Post announcement to blog.
rake post_news          # Post announcement to rubyforge.
rake publish_docs       # Publish RDoc to RubyForge
rake redocs             # Force a rebuild of the RDOC files
rake release            # Package and upload the release to rubyforge.
rake repackage          # Force a rebuild of the package files
rake ridocs             # Generate ri locally for testing
rake test               # Run the test suite.
rake test_deps          # Show which test files fail when run alone.
rake uninstall          # Uninstall the package.
```

newgem

newgem

- hoe wrapper
  - Non-viral: removes gem-consumer dependency on hoe
- adds a few extra tasks
- Supports RSpec and Test::Unit

newgem: bootstrap a gem

```
$ newgem object_identifier --test-with=rspec
      ...
      create  config
      create  doc
      create  lib
      ...
  dependency  install_rspec
      create    spec
      ...
Important
=========

* Open config/hoe.rb
* Update missing details (gem description, dependent gems, etc.)
```

newgem: installed rake tasks

```
$ cd object_identifier
$ rake -T
(in /Users/david/Documents/QCon/under_the_hood/gems/object_identifier)
rake announce             # Generate email announcement file and post to rub...
rake audit                # Run ZenTest against the package
rake check_manifest       # Verify the manifest
rake clean                # Clean up all the extras
rake clobber_docs         # Remove rdoc products
rake clobber_package      # Remove package products
rake config_hoe           # Create a fresh ~/.hoerc file
rake debug_gem            # Show information about the gem.
rake default              # Run the default tasks
rake deploy               # Release the website and new gem version
rake docs                 # Build the docs HTML Files
rake email                # Generate email announcement file.
...
```

newgem: installed rake tasks

```
...
rake gem                  # Build the gem file object_identifier-0.0.1.gem
rake generate_key         # Generate a key for signing your gems.
rake install              # Install the package.
rake install_gem          # Install the package as a gem
rake install_gem_no_doc   # Install the package as a gem, without generating...
rake local_deploy         # Runs tasks website_generate and install_gem as a...
rake manifest:refresh     # Recreate Manifest.txt to include ALL files
rake multi                # Run the test suite using multiruby
rake package              # Build all the packages
rake post_blog            # Post announcement to blog.
rake post_news            # Post announcement to rubyforge.
rake publish_docs         # Publish RDoc to RubyForge
rake redocs               # Force a rebuild of the RDOC files
rake release              # Package and upload the release to rubyforge.
rake repackage            # Force a rebuild of the package files
rake ridocs               # Generate ri locally for testing
...
```

newgem: installed rake tasks

```
...
rake spec                 # Run the specs under spec/models
rake test                 # Run the test suite.
rake test_deps            # Show which test files fail when run alone.
rake uninstall            # Uninstall the package.
rake website              # Generate and upload website files
rake website_generate     # Generate website files
rake website_upload       # Upload website files to rubyforge
```

Mr. Bones

bones

- A handy tool that builds a skeleton for your new Ruby projects
  - http://codeforpeople.rubyforge.org/bones/
- Supports RSpec and Test::Unit
- Lighter-weight than hoe or newgem
  - No dependencies
  - Non-viral
  - No tasks for publishing to RubyForge
  - No tasks for building a project website

bones: bootstrap a gem

```
$ bones object_identifier
created 'object_identifier'
now you need to fix these files
(in /Users/david/Documents/QCon/under_the_hood/gems/object_identifier)
README.txt:
  * [  2] [FIXME] (your name)
  * [  3] [FIXME] (url)
  * [  7] [FIXME] (describe your package)
  * [ 11] [FIXME] (list of features or problems)
  ...

Rakefile:
  * [ 13] [FIXME] (who is writing this software)'
  * [ 14] [FIXME] (your e-mail)'
  * [ 15] [FIXME] (project homepage)'
```

bones: installed rake tasks

```
$ cd object_identifier
$ rake -T
(in /Users/david/Documents/QCon/under_the_hood/gems/object_identifier)
rake clobber           # Remove all build products
rake doc               # Alias to doc:rdoc
rake doc:rdoc          # Build the rdoc HTML Files
rake doc:release       # Publish RDoc to RubyForge
rake doc:rerdoc        # Force a rebuild of the RDOC files
rake doc:ri            # Generate ri locally for testing
rake gem               # Alias to gem:package
rake gem:debug         # Show information about the gem
rake gem:gem           # Build the gem file object_identifier-0.0.0.gem
rake gem:install       # Install the gem
rake gem:package       # Build all the packages
rake gem:release       # Package and upload to RubyForge
rake gem:repackage     # Force a rebuild of the package files
rake gem:uninstall     # Uninstall the gem
```

newgem: installed rake tasks

```
...
rake manifest          # Alias to manifest:check
rake manifest:check    # Verify the manifest
rake manifest:create   # Create a new manifest
rake notes             # Enumerate all annotations
rake notes:fixme       # Enumerate all FIXME annotations
rake notes:optimize    # Enumerate all OPTIMIZE annotations
rake notes:todo        # Enumerate all TODO annotations
rake spec              # Alias to spec:run
rake spec:rcov         # Run all specs with RCov
rake spec:run          # Run all specs with basic output
rake spec:specdoc      # Run all specs with text output
rake test              # Alias to test:run
rake test:rcov         # Run rcov on the unit tests
rake test:run          # Run tests for run
```

Resources

- http://www.rubygems.org/
- http://rubyforge.org
- http://doc.rubygems.org
- http://blog.davidchelimsky.net
- http://blog.davidchelimsky.net/files/under_the_hood.zip
  - slides and code for this presentation