GMOペパボ

How DSL works on Ruby

The future of Rake

Hiroshi SHIBATA / GMO PEPABO inc.

2016.09.08 RubyKaigi 2016

GMOペパボ



Hiroshi SHIBATA @hsbt

Chief Engineer

https://www.hsbt.org

self.introduce

=>
{
  name: “SHIBATA Hiroshi”,
  nickname: “hsbt”,
  title: “Chief engineer at GMO Pepabo, Inc.”,
  commit_bits: [“ruby”, “rake”, “rubygems”, “rdoc”, “psych”, “syck”, “ruby-build”, “railsgirls”, “railsgirls-jp”, “tdiary”, “hiki”…],
  sites: [“hsbt.org”, “ruby-lang.org”, “rubyci.org”, “railsgirls.com”, “railsgirls.jp”],
}



My work in Ruby 2.4

- Maintain *.ruby-lang.org. Applied to Site Reliability Engineering.
- Gemify standard library (xmlrpc, tk)
- Update bundled library such as rubygems, rdoc, psych, json
- Maintain ruby-build
- Maintain Psych 2.0, 2.1, also Syck
- Maintain RDoc 4.2, 5.0
- Maintain Rake 10.5, 11.2 and 12.0(TBD)



1.

Rake

Make in Ruby

Rake is a Make-like program implemented in Ruby. Tasks and dependencies are specified in standard Ruby syntax.

```
task :awesome do
  puts :bar
end

task beat: [:awesome] do
  puts :buzz
end

task default: :beat
```



Rake features

- Rakefiles (rake's version of Makefiles) are completely defined in standard Ruby syntax.
- Users can specify tasks with prerequisites.
- Supports parallel execution of tasks.

```
$ rake -j
```



Rake::FileList

- It’s also declaration named `Filelist` on Toplevel.
- It has utility class for File listing.

```
file_list = FileList.new('lib/**/*.rb', ‘test/test*.rb')

FileList['a.c', 'b.c'].exclude("a.c") => [‘b.c']
```



Rake::TestTask

Minitest and Test::Unit integration task of Rake

```
require 'rake/testtask'

Rake::TestTask.new(:test) do |t|
  t.libs << "test"
  t.verbose = true
  t.test_files = FileList['test/**/test_*.rb']
end
```

It is only task for outside library. Task for rdoc, bundler, and more is stored their gems.



First SemVer

- Rake is a first famous gem adopted semantic versioning in rubygems
  - probably…
- Rake 0.9 bump to 10.0 for next version.
- Rake follows this release policy now.



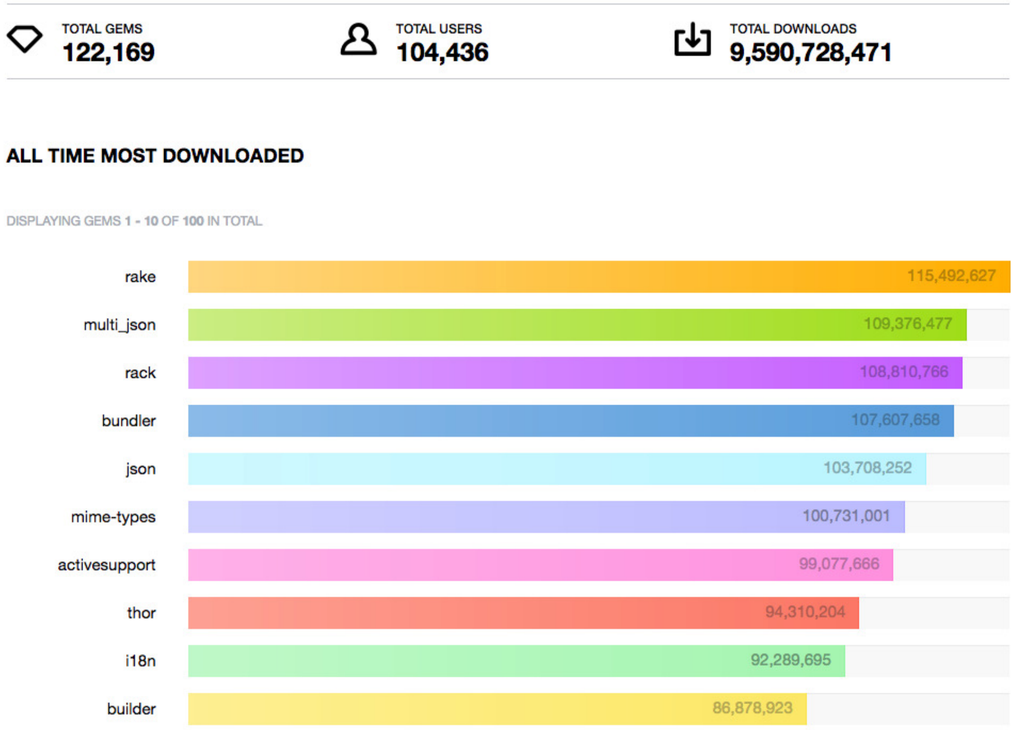
ruby/rake

- Rake was originally created by Jim Weirich, who unfortunately passed away in February 2014.
- This repository was originally hosted at github.com/jimweirich/rake, It has been moved to ruby/rake by @drbrain
- @drbrain and @hsbt maintain ruby/rake



Download Ranking

Rake is most downloaded gem in the ruby world.



2.

What’s DSL

DSL is Domain Specific Language

“A domain-specific language (DSL) is a computer language specialized to a particular application domain”

https://en.wikipedia.org/wiki/Domain-specific_language

In the situation of Ruby language

- **Ruby is readable language. Because we often use internal DSL.**
- **Ruby has a lot of functions for building internal DSL. It uses meta-programming technic.**



Pattern: Class method

Slightly simple DSL on standalone class

```
class User
  has_many :foo
end
```



Pattern: Class method

Slightly simple DSL on standalone class

```
class User
  has_many :foo
end
```

```
class User
  def self.has_many(foo)
    puts foo
  end
end
```



Pattern: Module and Class ancestors

You can build simple DSL used Ruby's module and class

```
class User < ARBase
  has_many :blogs
end
```



Pattern: Module and Class ancestors

You can build simple DSL used Ruby's module and class

```
class User < ARBase
  has_many :blogs
end
```

```
module DSL
  def has_many(bar = nil)
    puts bar
  end
end

class ARBase
  extend DSL
end
```



Pattern: Method define

You can define method via eval for simple DSL

```
class User < ARBase
  has_many :blogs
  blogs_foo
end
```



Pattern: Method define

You can define method via eval for simple DSL

```
class User < ARBase
  has_many :blogs
  blogs_foo
end
```

```
module DSL
  def has_many(bar = nil)
    self.class.module_eval <<-CODE, __FILE__, __LINE__
      def #{bar}_foo
        puts :foo
      end
CODE
  end
end
(snip)
```



Pattern: Implicit code block

We need to prepare to configuration for gem behavior changes.

```
Foo.configure do |c|
  c.bar = :buzz
end
```



Pattern: Implicit code block

We need to prepare to configuration for gem behavior changes.

```
Foo.configure do |c|
  c.bar = :buzz
end
```

```
module Foo
  def self.configure
    yield self
  end

  class << self
    attr_accessor :bar
  end
end
```



Pattern: Decrative setter

We can implement `Explicit code block` used instance_eval.

```
Foo.configure do
  bar :buzz
end
```



Pattern: Decrative setter

We can implement `Explicit code block` used instance_eval.

```
Foo.configure do
  bar :buzz
end
```

```
module Foo
  def self.configure(&block)
    instance_eval(&block)
  end

  class << self
    def bar(v = nil)
      v ? @bar = v : @bar
    end
  end
end
```



Pattern: instance_eval

You can provide class scoped DSL via instance_eval

```
gemfile = <<-GEM
  gem 'foo'
GEM

Foo.new.eval_gemfile(gemfile)
```



Pattern: instance_eval

You can provide class scoped DSL via instance_eval

```
gemfile = <<-GEM
  gem 'foo'
GEM

Foo.new.eval_gemfile(gemfile)
```

```
class Foo
  def gem(name)
    p name
  end

  def eval_gemfile(file)
    instance_eval(file)
  end
end
```



3.

DSL in Rubygems

DSL in Ruby language

- Rake
- Capistrano
- Thor
- Bundler



Rake and Rake.application

- rake_module.rb: defined `Rake.application` for Rake singleton instance.
- Rake.application returns `Rake::Application` instance.
- `rake` command invoke `Rake.application.run`

```
def run
  standard_exception_handling do
    init
    load_rakefile
    top_level
  end
end
```



Rake::Application#init , Rake::Application#load_rakefile

- `init` method handles…
  - detect invoking task: “rake -j foo” -> detecting “foo”
  - parse given option: “rake -j foo” -> detecting “-j”

- `load_rakefile` read default rakefile named `%w(rakefile Rakefile rakefile.rb Rakefile.rb)`



Rake::Application#top_level

- If you add `-T` options, top_level shows list of tasks on Rakefile
- If you add `-P` options, top_level shows dependencies of target task.

- top_level invokes tasks given command line. Example for `rake foo bar buzz`.
  top_level invokes three tasks.
- top_level methods are under the thread pool on Rake.



Object#task

- load_rakefile simply load Rakefile used by `load` method provided by Ruby.
- Your Rakefile DSL provided by `Rake::DSL` filed dsl_definition.rb

```
module Rake
  module DSL
    (snip)
    def task(*args, &block) # :doc:
      Rake::Task.define_task(*args, &block)
    end
  end
end
self.extend Rake::DSL
```



Rake::Task

- `Rake::Task` instance is defined by `Rake::TaskManager#define_task`
- `Rake::Task` have manipulation tasks on class methods like clear, tasks, [], and more.

- Class method of `Rake::Task` call `Rake::TaskManager` via `Rake.application` instance.



Rake::Task

- `Rake::TaskManager#define_task` build actions, dependencies and scope for task. actions are stored code blocks.

```
def enhance(deps=nil, &block)
  @prerequisites |= deps if deps
  @actions << block if block_given?
  self
end
```

- `Rake::Task#invoke` invoke tasks of dependencies and @actions for Proc#call
- Invoked task marked @already_invoked flag. If you clear this flag, you need to run `Rake::Task#reenable`



Capistrano

- Capistrano is a framework for building automated deployment scripts.
- Capistrano tasks on version 3 are simple rake task

```
task :restart_sidekiq do
  on roles(:worker) do
    execute :service, "sidekiq restart"
  end
end
after "deploy:published", "restart_sidekiq"
```

- `Capistrano::Application` inherited `Rake::Application`



Thor

Thor is a simple and efficient tool for building self-documenting command line utilities.

```
class App < Thor
  desc "list [SEARCH]", "list all of the available apps, limited by SEARCH"
  def list(search="")
    # list everything
  end
end
```

Thor provides inherited pattern DSL for their CLI. You can invoke it used `Thor.start`



Bundler

- `Bundler::CLI` inherited Thor class. It's simple thor command.
- Bundler provides Gemfile DSL used DSL class and instance_eval

```
def eval_gemfile(gemfile, contents = nil)
      expanded_gemfile_path = Pathname.new(gemfile).expand_path
      original_gemfile = @gemfile
      @gemfile = expanded_gemfile_path
      contents ||= Bundler.read_file(gemfile.to_s)
      instance_eval(contents.dup.untaint, gemfile.to_s, 1)
(snip)
```



CM



最新ニュース　セミナー・イベント情報　IRニュース

2016.09.08 プレスリリース minne

GMOペパボが運営する国内最大のハンドメイドマーケット「minne（ミンネ）」 Apple Payへの対応開始について

グローバルサイン認証サイト
SSL secured クリックして確認
GlobalSign by GMO

https://pepabo.com

もっとおもしろくできる



9:00 p.m.

ネットショッピングの支払いをする。

Safariでお気に入りのウェブサイトを開いてショッピングをする時も、あなたのiPhoneまたはApple Watchで支払いが完了します。アカウントを作る必要はありません。

http://www.apple.com/jp/apple-pay/



https://www.youtube.com/watch?v=BdQVd1uybCg



We are hiring!!1

Please follow our account @pb_recruit

4.

Long live the Rake

Rake 10.x

- My first maintained version of Rake
- I triaged issues and fixed broken test at first



JRuby compatible issues

- `Dir.chdir` change behavior of `sh()` on JRuby
  - https://github.com/ruby/rake/pull/101
  - https://github.com/jruby/jruby/issues/3653
  - https://github.com/hsbt/rake-issue-chdir



Reproduction code

```
def sh(*cmd)
  res = system(*cmd)
  status = $?
  p [res, status]
end

RUBY = ENV['RUBY'] || File.join(
    RbConfig::CONFIG['bindir'],
    RbConfig::CONFIG['ruby_install_name'] + RbConfig::CONFIG['EXEEXT'])

ENV['RAKE_TEST_SH'] = 'someval'

sh RUBY, 'check_no_expansion.rb', '$RAKE_TEST_SH', 'someval'

Dir.chdir 'test'
sh RUBY, 'check_no_expansion.rb', '$RAKE_TEST_SH', 'someval'
```



Results - Ruby 2.3.0

```
% ruby rake_issue.rb
["$RAKE_TEST_SH", "someval"]
[true, #<Process::Status: pid 24928 exit 0>]
["$RAKE_TEST_SH", "someval"]
[true, #<Process::Status: pid 24929 exit 0>]
```



Results - JRuby 9.0.5.0 and 1.7.24

JRuby 9.0.5.0

```
% ruby rake_issue.rb
["$RAKE_TEST_SH", "someval"]
[true, #<Process::Status: pid 21844 exit 0>]
["someval", "someval"]
[false, #<Process::Status: pid 21845 exit 1>]
```

JRuby 1.7.24

```
% ruby rake_issue.rb
["$RAKE_TEST_SH", "someval"]
[true, #<Process::Status: pid=22397,exited(0)>]
["$RAKE_TEST_SH", "someval"]
[true, #<Process::Status: pid=22398,exited(0)>]
```



Another compatible issues on JRuby

- `Exception#cause` is difference behavior with CRuby 2.3.0
  - https://github.com/jruby/jruby/issues/3654

- Missing stdout using `Open3.popen` after `Dir.chdir`
  - https://github.com/jruby/jruby/issues/3655



5.

Rake 11

Rake 11.x

- My first major release version of Rake
- It have some of breaking changes
- It uses modern ecosystem on ruby language



Removed deprecated code

- I removed deprecated code commented by Jim

- Some historical gems like yard and rspec used `TaskManager#last_comment`
- I reverted to delete `last_comment` at Rake 11
- I'm sorry to inconvenience experience for above breaking changes.



Rewrite hoe to bundler

- You may invoke to `bundle` when to see Gemfile
- Without bundler, you need to install hoe and their plugins.
- I sometimes rewrite gem release tasks at rake, psych, syck…etc.



Unexpected behavior for rake - verbose

- I misunderstand to behavior of `verbose` option on Rake#Testtask

- expected: verbose option of rake task same as Ruby's `-W` option. So you can get additional debug/warn message with your ruby code.
- actual: verbose option on rake displays details of command runner. and minitest has another `verbose` option. It shows test name and results for test suites.



Unexpected behavior for rake - deps

I added deps option for `Rake::TestTask`

```
task :setup do
end
Rake::TestTask.new do |t|
  t.deps = :setup
end
```

But It's same behavior this.

```
task :setup do
end

Rake::TestTask.new(:test) => [:setup] do
end
```



6.

Rake 12

Rake 12.x?

- I works to develop to Rake 12
- Release date is tentative
- This is major version-up. I have plan to some breaking changes.



Drop to support old Ruby

- Rake have concurrent task runner
- Current implementation detects core number used by system utilities like `sysctl`.
- But Ruby 2.2 provides `Etc.nprocessor` for detects core number natively.
- I like ruby core function. Rake 12 only supports Ruby 2.2 or later same as Rails 5.



minimize/minirake

- Rake have a lot of functions. I hope to reduce code for fast invocation when We run rake task.

- Idea 1: Reduce code without core function like `rake-contrib`
- Idea 2: minirake - mruby have minimum implementation of rake



default task

I will add default method for difine default task

```
task :default => [:foo, :bar]
```

to

```
default [:foo, :bar]
```



Do not use main(Object class)

- Rake defined task method under main instance provided the Object class.
- We cant use name of `task` on irb/rails console when you use Rake gem.
- I hope to solve it.



Conclusion

- Summarize Rake history and basis.
- Introduce DSL pattern of Ruby language
- Learn Rake internal
- Show Rake 10 and 11 works
- Propose Rake 12 future.



Let’s fan to maintain big OSS