

Managing Ruby gems with Bundler

http://github.com/igal/bundler_tutorial

Igal Koshevoy, Pragmaticcraft
Business-Technology Consultant
igal@pragmaticcraft.com
@igalko on Twitter & Identi.ca

Why did I care?

Wanted be able to deploy Rails app and its gems without any further interaction.

Did not want fuss of `gem` changing its behavior between versions, ugly user-installed gems trickery, passwords, `sudo rake gems:install`, etc.

What is Bundler?

- “Bundler is a tool that manages gem dependencies for your ruby application. It takes a gem manifest file and is able to fetch, download, and install the gems and all child dependencies specified in this manifest.”
- <http://github.com/carlhuda/bundler/>

Why you should care about Bundler?

- It's the way Rails 3 manages gems, no more `config.gem`
- You can easily use it from any Ruby app
- Easily install gems as a normal user, no more `sudo rake gems:install`
- Will likely make `rip` (Ruby's Intelligent Packaging) rest in peace, a pity
- Less fussy than `gem --user-install` and easier to access the environment

How to install Bundler?

```
# Bundler requires 1.3.6 or above  
sudo gem update --system
```

```
# Install Bundler itself  
sudo gem install bundler
```

```
# Yup, we've got Bundler now  
bundle -v
```

How to make a Bundler manifest?

Create "Gemfile" in directory w/:

A gem
gem "delorean"

A gem with a specific version
gem "delorean", "0.1"

A gem with a version equal to
or greater than 0.1
gem "delorian", "> 0.1"

A gem with a version equal to
or greater than 0.1 but less than 0.2
gem "delorean", "~> 0.1"

...continued from left...

A gem whose name and require
aren't the same, e.g.
`require "aws/s3"` rather than
`require "aws-s3"`
gem "aws-s3", :require => "aws/s3"

Groups
gem "rspec", :group => :test

group :test do
 gem "webrat"
end

How to use Bundler's manifests?

Check manifest

bundle check

Install what's needed

bundle install

Lock versions

bundle lock

How to use Bundler from plain Ruby app?

Gemfile

```
gem "delorean"
```

Ruby

```
require 'rubygems'  
require 'bundler'  
Bundler.require
```

```
# The library is there!  
p Delorean
```


How to use Bundler from Rails 3?

It's already there:

- `rails myapp`
- `cd myapp`
- `cat Gemfile`
- `bundle install`
- `script/rails console`

How to use Bundler from Rails 2?

```
### Append to "config/boot.rb"  
# Bundler integration from http://gist.github.com/302406 with  
# module/class hack for use within Passenger  
module Rails  
  class Boot  
    def run  
      load_initializer  
      extend_environment  
      Rails::Initializer.run(:set_load_path)  
    end  
  
    def extend_environment  
      Rails::Initializer.class_eval do  
        old_load = instance_method(:load_environment)  
        define_method(:load_environment) do  
          Bundler.require :default, Rails.env  
          old_load.bind(self).call  
        end  
      end  
    end  
  end  
end  
end  
end
```

"Managing Ruby gems with Bundler" for pdxruby - Igal Koshevoy - 2010-03-02

```
### Add to "config/preinitializer.rb"
# Bundler integration from http://gist.github.com/302406
begin
  # Require the preresolved locked set of gems.
  require File.expand_path('../../.bundle/environment', __FILE__)
rescue LoadError
  # Fallback on doing the resolve at runtime.
  require "rubygems"
  require "bundler"
  if Gem::Version.new(Bundler::VERSION) <= Gem::Version.new("0.9.5")
    raise RuntimeError, "Bundler incompatible.\n" +
      "Your bundler version is incompatible with Rails 2.3 and an unlocked bundle.\n" +
      "Run `gem install bundler` to upgrade or `bundle lock` to lock."
  else
    Bundler.setup
  end
end
```

How to deploy with Bundler?

Add to "config/deploy.rb"

```
namespace :deploy do
```

```
  ...
```

```
  desc "Install gems"
```

```
  task :gems, :roles => :app do
```

```
    run "cd #{release_path} && \
```

```
      (bundle check || bundle install) && bundle lock"
```

```
  end
```

```
end
```

```
...
```

```
after "deploy:finalize_update", "deploy:gems"
```

Gotchas? You betcha!

- Requires scary, brand-new version of RubyGems
- Bundler is brand-new code and has rough spots
- Rails 2 integration is ugly
- Passenger requires locked environment?
- Don't want to mix-and-match gem managers (gemrc, rip, etc)
- Can't easily require gems outside manifest on the fly
- ``bundle install`` accesses network even though it shouldn't, consider using ``bundle check || bundle install``
- ``bundle check`` checks everything, doesn't accept ``--with=GROUP`` or ``--without=GROUP``
- ``bundle install`` accepts ``--without=GROUP``, but not ``--with=GROUP``
- ``bundle check`` rewrites the locked environment file when checked even if same, annoying for versioning
- `#group` only allows one environment, unlike old
`if %w[test development].include?(RAILS_ENV); config.gem...`

Conclusion?

- It's useful
- It's wonky
- Start playing with it
- Maybe hold off on using it for another month or two for RubyGems and Bundler to stabilize some more
- ???