

The best way to manage your
application's dependencies

Bundler



The best way to manage your
application's dependencies

Bundler



Why Bundler?

applications need gem manifests

rake gems:unpack could cause LoadErrors

```
Gem::LoadError: can't activate rack (~> 1.0.0., runtime) for  
["actionpack-2.3.5"], already activated rack-1.1.0 for ["thin-1.2.7"]
```

rake gems:install could change gem versions

monkeypatches are a crappy way to fix bugs

Solutions

applications need gem manifests

resolve dependencies before installing

lock versions at install-time

allow gems to be used directly from git repos

Gemfile

```
source "http://rubygems.org"  
gem "rails", "3.0.0.beta3"  
gem "nokogiri"
```

```
$ bundle install
Fetching source index from http://rubygems.org/
Using rake (0.8.7)
Using abstract (1.0.0)
Installing builder (2.1.2)
Installing i18n (0.3.7)
Installing memcache-client (1.8.3)
Installing tzinfo (0.3.20)
Installing activesupport (3.0.0.beta3)
Installing activemodel (3.0.0.beta3)
Using erubis (2.6.5)
Using rack (1.1.0)
Installing rack-mount (0.6.3)
Using rack-test (0.5.3)
Installing actionpack (3.0.0.beta3)
Using mime-types (1.16)
Using polyglot (0.3.1)
Installing treetop (1.4.5)
Installing mail (2.2.0)
Using text-hyphen (1.0.0)
Using text-format (1.0.0)
Installing actionmailer (3.0.0.beta3)
Installing arel (0.3.3)
Installing activerecord (3.0.0.beta3)
Installing activeresource (3.0.0.beta3)
Using bundler (0.9.25)
Installing nokogiri (1.4.1) with native extensions
Installing thor (0.13.6)
Installing railties (3.0.0.beta3)
Installing rails (3.0.0.beta3)
Your bundle is complete!
Use `bundle show [gemname]` to see where a bundled gem is installed.
```

Gem management

`bundle install`

`bundle update`

`bundle pack`

`bundle install vendor`

Gem forks

Git gems

```
gem "rails", :git =>  
  "git@github.com:indirect/rails.git"
```

Path gems

```
gem "rails", :path => "~/sw/gems/rails"
```

Gem environments

```
group :development do
  gem 'rails-footnotes'
end
```

```
group :test do
  gem 'rspec'
end
```

```
group :production do
  gem 'pg'
end
```

```
$ bundle install --without production
```

Pure Ruby usage

```
source :gemcutter  
gem "nokogiri"
```

```
group :test do  
  gem "rspec"  
end
```

Pure Ruby usage

```
require "rubygems"  
require "bundler/setup"  
  
require "nokogiri"  
html = Nokogiri::HTML(%{  
  <h1>Hi!</h1>  
})
```

Pure Ruby usage

```
require "rubygems"  
require "bundler/setup"
```

```
Bundler.require  
html = Nokogiri::HTML(%{  
  <h1>Hi!</h1>  
})
```

Pure Ruby usage

```
require "rubygems"  
require "bundler"  
Bundler.setup(:default, :test)  
  
require "spec"
```

Gem development

```
require "rubygems"  
require "bundler"
```

```
Gem::Specification.new do |s|  
  s.name      = "Awesome"  
  s.author    = "Me"  
  s.version   = "1.0.0"  
  s.summary   = "My Awesome Gem"  
  
  s.add_dependency "json"  
  s.add_development_dependency "rspec"  
end
```

Gem development

```
source :gemcutter
```

```
gem "json"
```

```
group :development do  
  gem "rspec"  
end
```

```
group :other do  
  gem "ruby-debug"  
end
```

Gem development

```
require "rubygems"  
require "bundler"
```

```
Gem::Specification.new do |s|  
  s.name      = "Awesome"  
  s.author    = "Me"  
  s.version   = "1.0.0.0"  
  s.summary   = "My Awesome Gem"  
  
  s.add_bundler_dependencies  
end
```

Rails 3

That's it.
You don't have to do anything.

Rails 3

Well, until Bundler 1.0:
`bundle exec cucumber`

Rails 2

- Add `preinitializer.rb`
- Tweak `boot.rb`
- Use ``bundle exec cucumber`` until 1.0
- Code available at gembundler.com

Sinatra

```
source :gemcutter  
gem "sinatra"
```

Sinatra

```
require "rubygems"  
require "bundler/setup"  
require "sinatra/base"
```

```
class MyApp < Sinatra::Base  
  get "/" do  
    "Hi!"  
  end  
end
```

```
run MyApp
```

Deploying

```
after "deploy:update_code" do
  run "cd #{release_path} && bundle install"
end
```

```
$ bundle lock # only Bundler 0.9
$ git add Gemfile.lock
$ git commit -m "Locked gems for deploy"
$ git push
$ cap production deploy
```

Bundler 1.0 (beta)

- Coming *very soon*, later today
- Please test (in *non-critical* environments)
- Release candidate when feature complete
- Final release when bug list hits zero

Locking

- New lockfile format allows easy merging
- Bundler.require loads gems in Gemfile order
- Locking happens automatically
- No more `bundle lock` command
- Update versions with `bundle update`

`bundle exec`

- Deprecated! Exec is no longer required
- Bundler defaults to using system gems
- Prompts for sudo password if needed
- If using app-isolated gems, just prepend `~/bundle/bin` to your path

Multi-Platform

- Still under development, but works
- Cached gems don't to work correctly yet
- Gemfile syntax for specifying platforms:

Multi-Platform

```
platform :ruby_18 do
  gem "system_timer"
  gem "ruby-debug", ">= 0.10.3"
end
```

```
platforms :ruby, :rbx do
  gem 'yajl-ruby'
  gem "nokogiri", ">= 1.4.0"
end
```

```
platform :jruby do
  gem "jruby-openssl"
end
```

Thanks!

Bundler documentation at
gembundler.com

Questions, bug reports, and
discussion are welcome online
#bundler on freenode
github.com/carlhuda/bundler/issues