ChiliProject - Feature # 522: Script to correctly configure bundler

Status:	Open	**Priority:**	Normal
Author:	Holger Just	**Category:**	
Created:	2011-07-14	**Assignee:**	
Updated:	2012-03-25	**Due date:**	
Remote issue URL:			
Affected version:			
Description:	There should be a script which interactively checks dependencies and asks the user how to configure bundler.		

History

2011-07-15 10:07 pm - Eric Davis

I don't know about this. I've always hated how complex our installation is, bundler will help out quite a bit once everything is fully moved over (plugins) and we've got the versions figured out.

2011-07-25 05:34 pm - Holger Just

- Assignee deleted (Holger Just)

The script was originally intended to check that we have

* a supported ruby version
* a supported version of rubygems and bundler

It then asks the user which groups she wants to install (possibly checking dependencies like compilers or headers or proposing matching packages from the systems package manager). After that, bundler would be invoked with the default Gemfile.

However that might not be a really good ideas as this script could become arbitrarily complex and could also not properly deal with the requirements of different systems. Thus it might be better to just improve our documentation to lead users to properly install Bundler and all required dependencies.

But we still have two things to consider here:

1. Bundler's original scope is to define exact dependencies of top of rubygems. That's why they recommend to commit the @Gemfile.lock@ into Git.
2. ChiliProject is intended to be run on a variety of systems

It has been shown that Bundler's Gemfile syntax isn't as expressive as we would like it to have. Also it's hard to define dependencies which suit everybody (e.g. rmagick on old systems (like Lenny) vs. new systems (like Fedora), mysql or sqlite versions also differ).

Thus, it might be feasible to use the Gemfile as a kind of launcher and include other system specific Gemfiles for users who want to install just their OS packages and don't want to have a compiler. That way, we could have one Gemfile for each distribution and somehow instruct bundler to use those. Users who are good with a compiler can use our common Gemfile

What do you think about that idea?

2011-07-25 10:56 pm - Eric Davis

Holger Just wrote:

> Thus, it might be feasible to use the Gemfile as a kind of launcher and include other system specific Gemfiles for users who want to install just their OS packages and don't want to have a compiler. That way, we could have one Gemfile for each distribution and somehow instruct bundler to use those. Users who are good with a compiler can use our common Gemfile
>
> What do you think about that idea?

Maybe, I'd have to see the complexity before I could make a decision. It's too bad Bundler doesn't let you remove and re-version gems. Then we could tell a Debian user to download Gemfile.debian and Gemfile.mysql into @config/bundlers@ and have it "just work for them". Maybe that is the way we go and just have the core Gemfile be as minimal as possible. I don't know...

2012-03-24 09:14 pm - John Yani

Just one suggestion for a gem installation rake task: do not include plugins' gemfiles in the main gemfile. It will generate an error:

> You cannot specify the same gem twice with different version requirements.

if plugin also requires the same gem as Chiliproject. Better approach would be just to use bundler command in each plugin directory with --path pointing to chiliproject/vendor/bundle.

Also posted to Redmine: http://www.redmine.org/issues/10474

2012-03-25 07:32 am - Felix Schäfer

John Yani wrote:

> Just one suggestion for a gem installation rake task: do not include plugins' gemfiles in the main gemfile. It will generate an error:
>
> > You cannot specify the same gem twice with different version requirements.
>
> if plugin also requires the same gem as Chiliproject.

This is a problem with bundler (and one others have too, see Point 3 in http://alexbcoles.com/code/2012/03/20/bundler-pain-points.html).

> Better approach would be just to use bundler command in each plugin directory with --path pointing to chiliproject/vendor/bundle.

I don't think this would write the gems need by the plugins to the Gemfile.lock, i.e. they wouldn't be available to the running app, so this won't work.