Redmine - Feature #14030

Allow plugins to put gems inside PluginGemfile

2013-05-12 04:13 - Harry Garrood

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Jean-Baptiste Barth	**% Done:**	0%
Category:	Plugin API	**Estimated time:**	0.00 hour
Target version:	2.6.0		
Resolution:	Fixed		

Description

Having plugins put their gem requirements in Gemfile means that doing anything which involves Bundler inside the plugin directory (eg rake redmine:plugins:test) breaks.

Could Redmine's own Gemfile be changed so that it looks for files names, say, PluginGemfile as well?

Associated revisions

Revision 13337 - 2014-08-13 11:07 - Jean-Baptiste Barth

Add ability to define plugins' gem dependencies in PluginGemfile (#14030).

History

#1 - 2013-05-12 04:14 - Harry Garrood

To clarify -- Bundler sees a Gemfile in the current directory (/path/to/redmine/plugins/redmine_foo), and assumes that it is the root of the project, which is not the case.

#2 - 2013-05-12 13:18 - Jean-Baptiste Barth

- Assignee set to Jean-Baptiste Barth

Obviously you shouldn't run redmine rake tasks directly inside plugins. Bundler is not broken inside plugins per se. As plugins cannot run without redmine core, commands should be run from redmine core and optionnally limit effects to a specific plugin (with NAME=redmine_foo for instance).

Renaming things in a non-standard fashion would break plugins possible integration as rubygems which is not better I think.

#3 - 2013-05-12 14:11 - Harry Garrood

At the moment you shouldn't, but it would certainly be nice (and be less confusing for plugin authors) to be able to.

I don't see how it would break possible integration with rubygems -- it certainly wouldn't be the first gemfile to not be called Gemfile:

- bundle install has a --gemfile=FILE switch to allow you to use files with different names.
- Here: https://github.com/grosser/fast_gettext/tree/master/gemfiles is an example of a repository which contains multiple Gemfiles so that CI tests can be run against each set of gems.

#4 - 2013-05-12 14:35 - Jean-Baptiste Barth

Yes, you're right, I spoke too soon. Only gemspec is required for rubygems integration, some gems don't even have a Gemfile and it works well.

I'd love to hear Jean-Philippe thoughts on this point.

#5 - 2013-05-13 23:26 - Jean-Philippe Lang

OK for loading PluginGemfile. Gemfile should still be loaded if it exists for compatibility.

#6 - 2013-11-20 10:49 - Robin Böning

I think this Issue can be closed, since the Gemfile also loads Gemfiles of plugins, right?

#7 - 2014-08-13 11:08 - Jean-Baptiste Barth

- Status changed from New to Closed

- Target version set to 2.6.0

- Resolution set to Fixed

Robin -: not really, the requested feature is actually to be able to avoid the "Gemfile" file which has a specific meaning for bundler, so you could run bundler commands inside plugins without having bundler cry.

The feature was added in r13337.

#8 - 2021-01-16 18:26 - Jigar Mehta

I am currently facing a similar problem.

- I'm working on a Redmine plugin, mainly adding tests.
  - Plugin location: plugins/toggl2redmine - it's detected correctly.
- I wish to use Pry for debugging while I write my tests.
- Here's what I've tried:
  - Created a Gemfile in the root of my plugin and ran `bundle install` in Redmine root.
  - Created a PluginGemfile in the root of my plugin and ran `bundle install` in Redmine root.
  - Installed "pry" globally with "gem install pry"

None of the above seem to make "pry" available in my test. Nothing works, except when I add "pry" directly into "REDMINE/Gemfile".

I add some "puts" in that Gemfile and found that the `Dir.glob` block that tries to load the plugin gemfiles is actually detecting my plugin's Gemfile or PluginGemfile correctly, but just that pry is not installed even after "eval_gemfile". Am I missing something? Here's how my Gemfile looks:

```
# frozen_string_literal: true

source 'https://rubygems.org'

# I've also tried specifying a version, but the results are the same.
gem 'pry', group: [:development, :test]
```

I've also tried doing a `Bundler.require(*Rails.groups)` in my `test_helper.rb`. Nothing works :(

Update: It works!

Finally, I think I figured it out. Since the plugin Gemfile is included from within a main Gemfile, it does not need a "source". So, removing the "source" line fixes the problem.