Ruby master - Bug #6124

remove the "spec-only gems" in Ruby 1.9.3 (was What is the purpose of "fake" gems in Ruby)

03/07/2012 10:50 PM - vo.x (Vit Ondruch)

Status:	Assigned		
Priority:	Normal		
Assignee:	hsbt (Hiroshi SHIBATA)		
Target version:			
ruby -v:	ruby 1.9.3p0 (2011-10-30) [x86_64-linux]	**Backport:**	2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN

Description

As I tried to point out in #6123, the "fake" gems which are distributed with Ruby breaks user's expectations. The following example should fail:

$ ruby --disable-gems -e "puts require('bigdecimal')"
true

However, it is not failing. Could you please enlighten me what is the purpose of fake gem then? Even if you install updated BigDecimal from rubygems.org, the bundled version will won unless you use "gem 'bidgecimal'" somewhere in the code. This makes no sense.

Don't take me wrong, I am big fan of gemified stdlib #5481, however this is not the way how it should be done.

Related issues:

Related to Ruby master - Bug #6123: Properly gemify BigDecimal	**Rejected**	**03/07/2012**
Related to Ruby master - Feature #6590: Dealing with bigdecimal, etc gems in...	**Assigned**	

History

#1 - 03/08/2012 01:53 AM - Anonymous

On Wed, Mar 07, 2012 at 10:50:51PM +0900, Vit Ondruch wrote:

> Issue #6124 has been reported by Vit Ondruch.
>
> ---
>
> Bug #6124: What is the purpose of "fake" gems in Ruby
> https://bugs.ruby-lang.org/issues/6124
>
> Author: Vit Ondruch
> Status: Open
> Priority: Normal
> Assignee: Eric Hodel
> Category:
> Target version:
> ruby -v: ruby 1.9.3p0 (2011-10-30) [x86_64-linux]
>
> As I tried to point out in #6123, the "fake" gems which are distributed with Ruby breaks user's expectations. The following example should fail:
>
> $ ruby --disable-gems -e "puts require('bigdecimal')"
> true
>
> However, it is not failing. Could you please enlighten me what is the purpose of fake gem then? Even if you install updated BigDecimal from rubygems.org, the bundled version will won unless you use "gem 'bidgecimal'" somewhere in the code. This makes no sense.
>
> Don't take me wrong, I am big fan of gemified stdlib #5481, however this is not the way how it should be done.

I think this works because stdlib load path is searched regardless of
gem activation or not. These gems are real, they're just installed in a
special location.

I'm not sure if this is a good thing, but it does maintain backwards
compatibility.

--
Aaron Patterson

http://tenderlovemaking.com/

#2 - 03/08/2012 05:38 AM - vo.x (Vit Ondruch)

Yes, the availability of gems in stdlib loadpath is wrong, since everything what is in loadpath is loaded prior gems. RubyGems loading mechanism is just fallback.

Regarding the backward compatibly, it comes into play only when you use the --disable-gems flag and the flag itself is not backward compatible, so I see no point to talk about backward compatibility at all.

BTW I am not sure what you are referring by real gems, since these are definitely not real gems. Even the spec files are not real spec files, just some mockup. They don't contain the #require_paths (yes, it is not needed since the gems are already in the load path), I doubt you cannot refer them in Gemfile and vendor them using "bundle package", etc.

#3 - 03/08/2012 05:42 AM - vo.x (Vit Ondruch)

Anonymní wrote:

> On Wed, Mar 07, 2012 at 10:50:51PM +0900, Vit Ondruch wrote:
>
>> Issue #6124 has been reported by Vit Ondruch.
>>
>> ---
>>
>> Bug #6124: What is the purpose of "fake" gems in Ruby
>> https://bugs.ruby-lang.org/issues/6124
>>
>> Author: Vit Ondruch
>> Status: Open
>> Priority: Normal
>> Assignee: Eric Hodel
>> Category:
>> Target version:
>> ruby -v: ruby 1.9.3p0 (2011-10-30) [x86_64-linux]
>>
>> As I tried to point out in #6123, the "fake" gems which are distributed with Ruby breaks user's expectations. The following example should fail:
>>
>> $ ruby --disable-gems -e "puts require('bigdecimal')"
>> true
>>
>> However, it is not failing. Could you please enlighten me what is the purpose of fake gem then? Even if you install updated BigDecimal from rubygems.org, the bundled version will won unless you use "gem 'bidgecimal'" somewhere in the code. This makes no sense.
>>
>> Don't take me wrong, I am big fan of gemified stdlib #5481, however this is not the way how it should be done.
>
> I think this works because stdlib load path is searched regardless of
> gem activation or not. These gems are real, they're just installed in a
> special location.
>
> I'm not sure if this is a good thing, but it does maintain backwards
> compatibility.
>
> --
> Aaron Patterson
> http://tenderlovemaking.com/

BTW it does not answer the original question :) If they are not in load path, they don't need to be gems and vice versa.

#4 - 03/08/2012 07:53 AM - Anonymous

On Thu, Mar 08, 2012 at 05:38:05AM +0900, Vit Ondruch wrote:

> Issue #6124 has been updated by Vit Ondruch.
>
> Yes, the availability of gems in stdlib loadpath is wrong, since everything what is in loadpath is loaded prior gems. RubyGems loading mechanism is just fallback.
>
> Regarding the backward compatibly, it comes into play only when you use the --disable-gems flag and the flag itself is not backward compatible, so I see no point to talk about backward compatibility at all.
>
> BTW I am not sure what you are referring by real gems, since these are definitely not real gems. Even the spec files are not real spec files, just some mockup. They don't contain the #require_paths (yes, it is not needed since the gems are already in the load path), I doubt you cannot refer them in Gemfile and vendor them using "bundle package", etc.

Can you enumerate your criteria for differentiating "fake" gems from
"real" gems?

This may not be true in 1.9.3, but my trunk installation certainly has
gem specifications for stdlib gems:

[aaron@higgins ruby (probes)]$ gem list psych

*** LOCAL GEMS ***

psych (1.2.2)
[aaron@higgins ruby (probes)]$ gem spec psych | head -10
--- !ruby/object:Gem::Specification
name: psych
version: !ruby/object:Gem::Version
version: 1.2.2
prerelease:
segments:

- 1
- 2
- 2 platform: ruby [aaron@higgins ruby (probes)]$

I'm also able to use bundle package:

[aaron@higgins omg]$ bundle package
Fetching source index for http://rubygems.org/
Using psych (1.2.2)
Using bundler (1.0.22)
Your bundle is complete! Use bundle show [gemname] to see where a bundled gem is installed.
Updating .gem files in vendor/cache

- psych-1.2.2.gem [aaron@higgins omg]$ cat Gemfile source 'http://rubygems.org'

gem 'psych', '= 1.2.2'
[aaron@higgins omg]$

Can you be more specific about the issues you're encountering? Thanks!

--
Aaron Patterson
http://tenderlovemaking.com/

#5 - 03/08/2012 08:17 AM - vo.x (Vit Ondruch)

Let's take json for example:

C:>gem list | find "json"
json (1.6.3, 1.5.4)

C:>ruby -rjson -e "puts JSON::VERSION"
1.5.4

RubyGems always loads the newest version if not specified otherwise, but it does not apply for gems which are already bundled in Ruby.

And the reason I am wondering is, because this bundling again makes hard time to package the bundled gems for Fedora, it makes hard to update them. All just because they don't behave like other gems, because they are not gem in fact. And if there is .gemspec file does not really matter IMO.

#6 - 03/08/2012 08:19 AM - vo.x (Vit Ondruch)

Just for comparison, I did the same for Arel, but I'm sure you can use other non-bundled gems:

C:>gem list | find "arel"
arel (2.2.1, 2.0.10)

C:>ruby -rarel -e "puts Arel::VERSION"
2.2.1

#7 - 03/08/2012 11:14 AM - nahi (Hiroshi Nakamura)

I wrote some existing issues of "fake gem" at https://bugs.ruby-lang.org/issues/5481#note-1 which is excerpted from
https://bugs.ruby-lang.org/projects/ruby/wiki/StdlibGem.
Please let us know at #5481 when you find additional issue of existing "fake gem". This must be fixed as "default gem", the word I introduced for expressing expected behavior, that is almost the same thing as "fake gem" :)

#8 - 03/08/2012 04:34 PM - vo.x (Vit Ondruch)

Hiroshi Nakamura wrote:

> I wrote some existing issues of "fake gem" at https://bugs.ruby-lang.org/issues/5481#note-1 which is excerpted from https://bugs.ruby-lang.org/projects/ruby/wiki/StdlibGem.
> Please let us know at #5481 when you find additional issue of existing "fake gem". This must be fixed as "default gem", the word I introduced for expressing expected behavior, that is almost the same thing as "fake gem" :)

Hiroshi,

I am aware of #5481, you can notice my comments there. However, the "fake gems" or "spec-only gems" how you call it are biting right now as I shown above.

So I would like to see (1) removed the "spec-only gems" in Ruby 1.9.3, since I can't see benefit in current state, or (2) make "default gems" from the "spec-only" gems right now for Ruby 1.9.3.

I prefer (2) and we actually do it already in Fedora 17 [1].

[1] http://pkgs.fedoraproject.org/gitweb/?p=ruby.git;a=blob;f=ruby.spec;hb=HEAD#l381

#9 - 03/14/2012 05:30 PM - Anonymous

Here is another issue:
If a gem package wants to depend on a gemified piece of standard library, it should also have it in its gemspec, right? As Vit has already mentioned, we had completely unbundled the fake-gems, but we are getting another issue: If the gemspec requires, for example, bigdecimal >= 1.1.0, it won't work with Ruby < 1.9, because bigdecimal gem doesn't work with these. So even if a new version of bigdecimal comes out and a gem developer wants to add it to his gemspec to make sure he uses it, he can't, because his package won't work with Ruby < 1.9 then.

#10 - 03/18/2012 06:46 PM - shyouhei (Shyouhei Urabe)

- Status changed from Open to Assigned

#11 - 04/19/2012 04:25 PM - vo.x (Vit Ondruch)

Just to illustrate, this is another issue with fake gems: https://github.com/rubygems/rubygems/issues/79#issuecomment-5214740

In short, RubyGems fails to generate documentation, since they cannot find the sources of BigDecimal where they are supposed to be.

#12 - 05/12/2012 03:58 AM - vo.x (Vit Ondruch)

Another troubling issue might be found at [ruby-core:44996]

#13 - 06/04/2012 02:32 PM - Anonymous

Activesupport gem seems to have a problem with this, too: https://github.com/rails/rails/issues/5355.
In short, if new version of bigdecimal is released, activesupport won't pick it up, as it is not in the gemspec and therefore the "fake gem", present in ruby load path will get picked up. So what exactly is the point of splitting the standard library into gems, when updating these gems doesn't actually put the new ones into use?

#14 - 06/13/2012 06:09 AM - headius (Charles Nutter)

A similar problem exists for JRuby.

We have not split bigdecimal out into a gem, so anyone that wants to add the gem as a dependency will fail on JRuby. We may need to work with ruby-core to push a dummy "bigdecimal" gem so that such specs will work.

#15 - 06/13/2012 10:44 AM - naruse (Yui NARUSE)

- Assignee changed from drbrain (Eric Hodel) to nobu (Nobuyoshi Nakada)

#16 - 06/13/2012 10:53 AM - nobu (Nobuyoshi Nakada)

- Status changed from Assigned to Rejected

#17 - 06/13/2012 11:00 AM - nahi (Hiroshi Nakamura)

- Assignee changed from nobu (Nobuyoshi Nakada) to nahi (Hiroshi Nakamura)

Assigned to me because this discussion seems related to stdlib gemification. I'll check this discussion before 2.0.

#18 - 06/13/2012 11:01 AM - nahi (Hiroshi Nakamura)

- Status changed from Rejected to Open

Make it open to keep reminding myself.

#19 - 06/13/2012 12:10 PM - nobu (Nobuyoshi Nakada)

- Status changed from Open to Closed

Sorry for close without the description.

The default gem just shows the bundled libraries as also gems, not avoid to install them as-is.
The --disable=gem option does not have any effect on standard libraries.

nahi (Hiroshi Nakamura), just keep in your bookmark.

#20 - 06/13/2012 12:23 PM - shyouhei (Shyouhei Urabe)

□□□-dev□□□□□□□

On 06/13/2012 10:53 AM, nobu (Nobuyoshi Nakada) wrote:

> Issue #6124 has been updated by nobu (Nobuyoshi Nakada).
>
> Status changed from Assigned to Rejected
>
> ---

□□□reject□□□□□□□□□□□□? □□□□□□□

□□□□□□□□□□□□□□□□□□□□□□

#21 - 06/13/2012 02:45 PM - Anonymous

nahi (Hiroshi Nakamura) wrote:

> Assigned to me because this discussion seems related to stdlib gemification. I'll check this discussion before 2.0.

I think that this deserves the attention now, as it's a conceptual problem. I believe that it won't be matter of hours to solve this, so I would advise doing this sooner rather than later. I'll be happy to throw any ideas and help out with the coding, if needed.
nobu (Nobuyoshi Nakada): We know what the behaviour is and how things work. What we are saying is, that it should be done in a different way :)

#22 - 06/14/2012 06:07 AM - naruse (Yui NARUSE)

nobu (Nobuyoshi Nakada) wrote:

> Sorry for close without the description.
>
> The default gem just shows the bundled libraries as also gems, not avoid to install them as-is.
> The --disable=gem option does not have any effect on standard libraries.
>
> nahi (Hiroshi Nakamura), just keep in your bookmark.

nobu's explanation is still too short, so I additional to say,

```
ruby -rjson -e "puts JSON::VERSION"
1.5.4
```

Strictly speaking, Vit's example showed in [ruby-core:43126] is not an issue of fake gem.
This is mainly because of the behavior of rubygems itself and require.

See following examples:

```
% ruby -rjson -e'puts JSON::VERSION'
1.5.4
% ruby -e'require"json";puts JSON::VERSION'
1.5.4
% ruby -e'gem"json";require"json";puts JSON::VERSION'
1.7.3
```

As they show, you can load gem's json if and only if call gem "json".
So what you want to do is already you can,
and if you want to write this shorter, it is another issue.

#23 - 06/14/2012 02:26 PM - vo.x (Vit Ondruch)

naruse (Yui NARUSE) wrote:

> nobu (Nobuyoshi Nakada) wrote:
> % ruby -rjson -e'puts JSON::VERSION'
> 1.5.4
> % ruby -e'require"json";puts JSON::VERSION'
> 1.5.4
> % ruby -e'gem"json";require"json";puts JSON::VERSION'
> 1.7.3
>
> As they show, you can load gem's json if and only if call gem "json".
> So what you want to do is already you can,
> and if you want to write this shorter, it is another issue.

Yes, you can, but you don't have to do it for other "real" gems. But that was not my only point. That was just one of many examples how the "fake" gems break expectations, such as ruby-core:44996 and ruby-core:45414. You are providing me workaround where I am looking for final solution.

#24 - 06/15/2012 05:30 AM - naruse (Yui NARUSE)

vo.x (Vit Ondruch) wrote:

> naruse (Yui NARUSE) wrote:
>
> > nobu (Nobuyoshi Nakada) wrote:
> > % ruby -rjson -e'puts JSON::VERSION'
> > 1.5.4
> > % ruby -e'require"json";puts JSON::VERSION'
> > 1.5.4
> > % ruby -e'gem"json";require"json";puts JSON::VERSION'
> > 1.7.3
> >
> > As they show, you can load gem's json if and only if call gem "json".
> > So what you want to do is already you can,
> > and if you want to write this shorter, it is another issue.
>
> Yes, you can, but you don't have to do it for other "real" gems. But that was not my only point.

Could you clarify your proposal?

> That was just one of many examples how the "fake" gems break expectations, such as ruby-core:44996 and ruby-core:45414. You are providing me workaround where I am looking for final solution.
>
> [ruby-core:44996]

It seems not an issue of fake gems.
It is only a bug of rake/testtask.rb.

> [ruby-core:45414]

I can't imagine the use case of this, and I don't think this should be fixed.

#25 - 06/22/2012 07:24 PM - vo.x (Vit Ondruch)

Hi,

I put together patch (available in my github [1] fork), which demonstrates on Rake, how the situation could be improved. Would you mind to review it and let me know if such approach would be acceptable? If yes, I am willing to work further that way.

The main point of that patch is that the gemified libraries are moved out of StdLib locations. This allows to clearly see that something is available as a gem. It also does not suffer from any of the symptoms mentioned above. The ./gems/rake directory I am proposing for Rake could be even some SVN/GIT submodule, so it could help to avoid the uncertainty where is the upstream source coming from.

[1] https://github.com/voxik/ruby/commits/move-gems/

#26 - 07/01/2012 07:35 AM - nahi (Hiroshi Nakamura)

- Subject changed from What is the purpose of "fake" gems in Ruby to remove the "spec-only gems" in Ruby 1.9.3 (was What is the purpose of "fake" gems in Ruby)

- Status changed from Closed to Open

Vit proposed either of these;
(1) remove the "spec-only gems" in Ruby 1.9.3
(2) make "default gems" from the "spec-only" gems right now for Ruby 1.9.3

Since (2) should wait for #5481 and 2.0 so (1) is the only option right now I think.

I'm positive for (1). Thoughts?

#27 - 07/14/2012 06:34 PM - mame (Yusuke Endoh)

- Status changed from Open to Assigned

#28 - 11/06/2018 04:58 PM - hsbt (Hiroshi SHIBATA)

- Assignee changed from nahi (Hiroshi Nakamura) to hsbt (Hiroshi SHIBATA)

Files

noname	500 Bytes	03/08/2012	Anonymous
noname	500 Bytes	03/08/2012	Anonymous