

Ruby - Bug #15354

Remove Bundler from StdLib

11/28/2018 05:11 PM - vo.x (Vit Ondruch)

Status:	Rejected	
Priority:	Normal	
Assignee:	hsbt (Hiroshi SHIBATA)	
Target version:		
ruby -v:	ruby 2.6.0dev (2018-11-26 trunk 65990) [x86_64-linux]	Backport: 2.4: UNKNOWN, 2.5: UNKNOWN
Description [This is mostly a clone of #13978 which I opened a year ago. Unfortunately, the same points I mentioned there still hold true.] I understand that given that almost every Ruby user is using Bundler, it would be convenient for quite some people to have Bundler as part of StdLib. However, seeing two copies of Molinillo, each in a different version 1, 2, similarly to fileutils 3, 4, is a sign that things are not as they should be. Therefore, please consider removal of Bundler from StdLib unless upstream demonstrates it can maintain it properly.		

History

#1 - 11/28/2018 05:13 PM - leam (Leam Hall)

Related? <https://bugs.ruby-lang.org/issues/15333>

#2 - 11/28/2018 05:41 PM - shevegen (Robert A. Heiler)

I personally do not use bundler, so I don't mind either way, but I use gems (and would love to preserve gems how I used them in the last ... 10 years or so; e. g. only using .gemspec rather than Gemfile).

I understand the discussion is a lot between maintaining the rubygem-code and functionality made available through bundler; drbrain mentioned this in the past on various issues at github, for the rubygem-page there.

I further assume that a lot of the rails users make use of bundler + Gemfile a lot; while I do not use rails myself either (it's surprising how much I don't use, but I DO use ruby just about daily), I think the net benefits outweigh the problems as described here. So I think it is not really a realistic suggestion to remove bundler, even more so after the efforts to integrate it.

I may be wrong, but I think I remember indirect commented that he is committed to iron out any potential problems in regards to bundler; and among the ruby core team, Hiroshi is the one doing a lot of work to make the integration happen.

I do not have enough insight knowledge to comment how bundler and gem co-operate with one another; and how it then interacts with the rest of ruby. However had, I think the ruby xmas release will be considered a stable build, so things not working there will probably not make it into the xmas release. As I myself am not affected I don't mind either way but I would hope that the bundler team would like to see bundler become an "official" part of ruby distributed by default. This would actually be a net win for those who DO use bundler.

There is still a month left so I'd give them a bit more time. Having fileutils duplicated is ... strange indeed, but I do not think this is intended, so I'd just be patient for the moment.

PS: We can already have multiple versions via gems, and I assume via bundler too. I don't know the bundler way (it is too complicated for me), but with gems "gem" always asks me if I want to uninstall something, and if there is more than one version, which version to remove, or whether to remove all of them. I think such functionality should be kept the same without duplicates. I assume that bundler itself may also become a bit simpler when it is a "first-class citizen" like gem, e. g. it could "ask" gem to uninstall something, and gem would know where bundler puts stuff too. Duplicates more sound like a bug; a bug that ought to be fixed, but this should not be confused with equaling it as a "removing bundler from stdlib" - that is not realistic, even if we may all agree that the quality could be improved.

In the event that bundler could never be merged, I don't see the big problem either - it could stay separate too, at the least as far as I am concerned. :) Gem suffices for my needs perfectly well (I manage everything on my own though, via ruby; including compiling things from source, tracking gems etc... so I have no need for bundler since my ruby scripts do something very similar in the end. I understand that different people aren't the same and have different use cases - I think what is forgotten in the issue tracker here, or the one from a year ago, is that it is indeed these ruby users that COULD benefit the most from bundler being part of ruby by default, are, in my opinion, the biggest pro-reason for trying to merge bundler into ruby).

Perhaps we could get one from the bundler team to also comment on it - it seems as if only Hiroshi comments but not the bundler core team? Do they not read the main issue tracker here?

#3 - 11/28/2018 10:53 PM - hsbt (Hiroshi SHIBATA)

- Status changed from Open to Assigned

- Assignee set to hsbt (Hiroshi SHIBATA)

#4 - 11/29/2018 05:51 AM - MSP-Greg (Greg L)

@vo.x

Many of the points you make are true.

I also think about when Travis mucked up their builds (at least trunk, maybe more) with RubyGems & Bundler, and at least a few popular repos/gems had CI issues. You mentioned that many users have Bundler installed, the same is also true regarding its use in CI.

Having contributed to RubyGems over the past year, I support the decision by [@hsbt \(Hiroshi SHIBATA\)](#) to add Bundler to trunk. Your issues will probably get resolved quicker with it included...

#5 - 11/29/2018 06:35 AM - vo.x (Vit Ondruch)

I am afraid that my issues won't be addressed as long as independent Bundler is not discontinued and this have not happened for past 10 years. If this was just announced, the two copies of Molinillo could be merged into one for start. The repositories could be rearranged to have more sense. The backward compatibility and extensions lib/bundler/rubygems_*.rb could be probably removed or slimmed down. But nothing like that is planned. I just hear rumors about Bundler 2.x and Bundler 3.x, which will necessarily keep carrying all these issues.

#6 - 11/29/2018 11:09 PM - colby (Colby Swandale)

vo.x (Vit Ondruch) wrote:

please consider removal of Bundler from StdLib unless upstream demonstrates it can maintain it properly.

Hi, Bundler core team dev here.

What does this even mean? We have been maintaining Bundler "properly", It's almost a 10 year project that has had to deal with lots of issues from the RubyGems/Ruby project over the years. I suggest you rethink/rescind your snide comment.

#7 - 11/30/2018 12:49 AM - MSP-Greg (Greg L)

[@colby \(Colby Swandale\)](#)

Ok. I decided on the longer message.

snide comment

Snide - 'derogatory or mocking in an indirect way'. @vo.x's comment is direct, and his point about the vendored libraries is valid.

(Bundler) has had to deal with lots of issues from the RubyGems/Ruby project

Something just seems backward...

Just a few minutes ago, I came across a test in RubyGems that had three nested unless/end pairs, and only the innermost had any code. So, I thought about a PR to clean it up; my first thought was who would ask 'what does this fix?'.

Also in RG, IMHO, there's code where process x and process y have become so tangled that they need to be pulled apart to allow other code to work correctly. The tangling also mucks up testing. I've procrastinated just because...

Conversely, here in ruby/ruby, the attitude is much more about improvement of any type.

Moving on, what are your thoughts on:

1. In what way should ruby/RubyGems/Bundler be integrated?
2. What to do about the vendored/duplicated libraries?
3. Given your involvement in both Bundler & RubyGems, and assuming infinite resources, what should change?

Thanks, Greg

#8 - 11/30/2018 01:01 AM - hsbt (Hiroshi SHIBATA)

- *Status changed from Assigned to Rejected*

I have the same concerns.

I'm going to resolve the duplicated vendoring problem at <https://github.com/rubygems/rubygems/pull/2026>.