Ruby - Misc #19421

Distribution documentation

02/07/2023 05:03 AM - ioquatix (Samuel Williams)

Status:	Open	
Priority:	Normal	
Assignee:		

Description

I use Ruby a lot, on a lot of different systems, and help people and companies use it, including developers who install it on their systems.

Over time, I found that installing Ruby isn't always easy. Part of this is due to package management. There are many systems, and Ruby has had some tricky migrations (e.g. OpenSSL is probably one of the most painful ones that lots of developers have trouble with).

Arch Linux has been stuck on Ruby 3.0 for a long time, which could be considered surprising given that Arch Linux is often on the bleeding edge of releases. I personally use Arch too. So I decided to ask, what is holding them up from making a release?

I found out they had many questions about how to distribute Ruby correctly. When I listened to those questions I felt that there are many ambiguities in how we build and package Ruby for operating system packages. This isn't to say that there isn't a good way to do it, just that we as a core team might be able to improve our communication about how Ruby is evolving and the implications for package managers (if any).

I've introduced doc/distribution.md as an effort to start having better documentation for people distributing Ruby. There are many ways which people distribute Ruby, and many "partial" documentation or assumptions being made about how to distribute Ruby and I'd like to provide a convenient standard location that can help people build package for Ruby distribution. Ultimately this makes my job easier because the latest versions of Ruby will be easier to install, so that's what I care about, but I don't care about what specifically is in the document, except that I think we should listen to the kinds of questions being asked and, in the best interest of Ruby, provide guidance.

There was a lot of good discussion on the PR, but my goal is not to make a finished document, but instead plant the seed so it can grow.

https://github.com/ruby/ruby/pull/6856

Some follow up discussion is required:

- What is the best practice for building source packages. The documentation I wrote from this was removed as "out of scope" but I disagree with that (https://github.com/ruby/ruby/commit/c35ebed895e1a3f7bced3db50ea0db8f284744e8). I don't have a strong opinion about what it should look like, but I think we should give a clear example of how to build source packages like what I wrote.
- Related to the above, what is the official location for source tarballs?
- What optional dependencies are required for building vs distributing Ruby. Arch Linux currently lists: doxygen gdbm graphviz libffi libyaml openssl ttf-dejavu tk as dependencies but it's not clear if this list is up to date, or what the expectations are. When someone installs Ruby (e.g. apt-get install ruby) what dependencies should be installed? I think it would be helpful to list expected dependencies (from a system package POV), etc.
- Is Ruby needed for building Ruby? Should source packages install Ruby before building from source? If so, what versions are supported?
- Clear guidance on gems that are distributed an alongside Ruby, and how security changes to gems are managed.

Even if we have more detailed documentation elsewhere, let's summarise it and then cross-reference it. People who build packages to distribute and install Ruby should feel supported, they are very important to our community. To this end, I established #distribution channel on Slack for discussion. We should listen to the questions being asked and use those questions to drive improvements to documentation.

History

#1 - 02/07/2023 05:04 AM - ioquatix (Samuel Williams)

- Tracker changed from Bug to Misc

- Backport deleted (2.7: UNKNOWN, 3.0: UNKNOWN, 3.1: UNKNOWN, 3.2: UNKNOWN)

#2 - 02/07/2023 05:23 AM - hsbt (Hiroshi SHIBATA)

> Related to the above, what is the official location for source tarballs?

See https://www.ruby-lang.org/en/downloads/.

> What optional dependencies are required for building vs distributing Ruby. Arch Linux currently lists: doxygen gdbm graphviz libffi libyaml openssl ttf-dejavu tk as dependencies but it's not clear if this list is up to date, or what the expectations are. When someone installs Ruby (e.g. apt-get install ruby) what dependencies should be installed? I think it would be helpful to list expected dependencies (from a system package POV), etc.

See https://github.com/ruby/ruby/blob/master/doc/contributing/building_ruby.md and https://github.com/rbenv/ruby-build/wiki#suggested-build-environment

> Is Ruby needed for building Ruby? Should source packages install Ruby before building from source? If so, what versions are supported?

Added it at https://github.com/ruby/ruby/commit/5ee39ea67f1e683322363129c82be83dea2432b1

> Clear guidance on gems that are distributed an alongside Ruby, and how security changes to gems are managed.

It's distributor's convenience, not our responsibility scope.

#3 - 02/07/2023 05:37 AM - jeremyevans0 (Jeremy Evans)

ioquatix (Samuel Williams) wrote:

> the best practice for building source packages. The documentation I wrote from this was removed as "out of scope" but I disagree with that (https://github.com/ruby/ruby/commit/c35ebed895e1a3f7bced3db50ea0db8f284744e8). I don't have a strong opinion about what it should look like, but I think we should give a clear example of how to build source packages like what I wrote.

As the maintainer of the Ruby port for OpenBSD, here's what I think.

The general build instructions should work:

```
configure # with appropriate options if needed
make
make install # as superuser if needed
```

Anything more than that should probably be handled in a per-operating system manner.

> - Related to the above, what is the official location for source tarballs?

https://cache.ruby-lang.org/pub/ruby/MAJOR.MINOR/

> - What optional dependencies are required for building vs distributing Ruby. Arch Linux currently lists: doxygen gdbm graphviz libffi libyaml openssl ttf-dejavu tk as dependencies but it's not clear if this list is up to date, or what the expectations are. When someone installs Ruby (e.g. apt-get install ruby) what dependencies should be installed? I think it would be helpful to list expected dependencies (from a system package POV), etc.

These libraries are needed for a full Ruby install:

- libffi: used for fiddle
- libyaml: used for psych
- libssl/libcrypto: used for openssl (provided by OpenSSL or LibreSSL)
- libreadline: use for readline (not really required anymore with reline)
- libgmp: used for faster Bignum calculations (maybe Rational as well)
- libz: used for zlib

For each, you should have both the headers and the shared/static libraries, if the operating system packages them separately.

gdbm and tk were removed as included extensions a while back. Not sure why ttf-dejavu would be needed, that looks like a font. doxygen and graphviz are looked for in the configure script, but I'm not sure to what extent they are used. They aren't needed for a normal Ruby install AFAIK.

I think this is more fully covered at https://docs.ruby-lang.org/en/master/contributing/building_ruby_md.html

- Is Ruby needed for building Ruby? Should source packages install Ruby before building from source? If so, what versions are supported?

Not when building from a tarball, unless the tarball has a bug (I think 3.2.0-preview2 did). Ruby is needed for building Ruby from the git repository, but that probably isn't something operating system packagers should be doing.

- Clear guidance on gems that are distributed an alongside Ruby, and how security changes to gems are managed.

All bundled and standard gems are included in the distribution tarball. How security changes to gems are managed seems a different discussion than how distribution should be handled, but it is related. In general, with either standard or bundled gems, you can gem install to get updated versions. However, for standard gems, the updated versions won't be used unless loaded via the gem method or when included in a Gemfile.

> Even if we have more detailed documentation elsewhere, let's summarise it and then cross-reference it. People who build packages to distribute and install Ruby should feel supported, they are very important to our community. To this end, I established #distribution channel on Slack for discussion. We should listen to the questions being asked and use those questions to drive improvements to documentation.

From my experience, packaging differs widely across operating systems. The job of an operating system packager is to make the package integrate better into the operating system, and how that is done can vary widely across operating systems. I think it's great to provide more support as needed, but we should make sure the documentation is helpful without being authoritarian. The documentation should answer questions that packagers may have, but it should not state definitively how something should be done by packagers.

#4 - 02/07/2023 06:25 AM - ioquatix (Samuel Williams)

> It's distributor's convenience, not our responsibility scope.

Open source is no one's responsibility and everyone's convenience.

I don't think this should prevent interested parties adding and maintaining documentation.

> https://www.ruby-lang.org/en/downloads/

Does the source package script need to consult this page to find the URL, or are the URLs on this page considered official?

i.e. are the URLs such as https://cache.ruby-lang.org/pub/ruby/MAJOR.MINOR/ on that page considered stable (i.e the URI and checksum) for long term use?

#5 - 02/07/2023 06:33 AM - hsbt (Hiroshi SHIBATA)

> The documentation should answer questions that packagers may have, but it should not state definitively how something should be done by packagers.

I totally agreed this.

#6 - 02/07/2023 06:39 AM - ioquatix (Samuel Williams)

> gdbm and tk were removed as included extensions a while back.

Was this ever communicated to downstream maintainers clearly? If not, is there a way we can improve that?

> Ruby is needed for building Ruby from the git repository, but that probably isn't something operating system packagers should be doing.

Some operating system packages do build directly from git ("source packages").

> How security changes to gems are managed seems a different discussion than how distribution should be handled, but it is related.

You are right it is a separate discussion (e.g. https://bugs.ruby-lang.org/issues/19178), but distros are going to want a clear picture of how that process works because for them, they are building and distributing packages and updates, and usually desire a certain level of stability, etc. Of course, this isn't a problem for us to solve, but clearly communicating what people could expect to receive downstream would be a big improvement.

In other words, we can document it once, or we can leave it up to chance and continue to answer the same questions over and over again.

#7 - 02/07/2023 06:40 AM - ioquatix (Samuel Williams)

> The documentation should answer questions that packagers may have, but it should not state definitively how something should be done by packagers.

I agree with this too, but I don't think that should prevent us from giving examples of HOW things can be done and pointing at downstream implementations.

#8 - 02/07/2023 10:01 AM - vo.x (Vit Ondruch)

hsbt (Hiroshi SHIBATA) wrote in #note-2:

> See https://github.com/ruby/ruby/blob/master/doc/contributing/building_ruby.md

Unfortunately, these documents are kind of useless. They explain what is needed but don't explain why and that is the problem. E.g.

- if I am not mistaken, Ruby can be built without OpenSSL. But what is the impact of disabling OpenSSL?
- use of GMP is not documented at all as far as I can say. What is the benefit of enabling it?
- Is DTrace/SystemTap documented somewhere?
- Why gperf? I am not aware I'd need this for anything creating the tarball via make-snapshot or using the tarball directly.
- Are these dependencies supposed to cover also execution of test suite?
- Not sure why there is mix of mentioning tarball and git checkout and what is the preferred method and why. Not mentioning that the build experience is vastly different from dependencies POV (yes, having Ruby installed or not is significant difference, having Ruby installed during build of Ruby package on Fedora is lets say problematic due to customizations in operating_system.rb).

> and https://github.com/rbenv/ruby-build/wiki#suggested-build-environment

I really wonder what is the source of this information? Just looking at Fedora, I don't understand why patch should be recommended there. And for CentOS / RHEL, why gcc-6? Also RHEL9 has more recent toolchain then RHEL6, so the instructions should be probably differentiated. I don't think that ncurses-devel is needed anymore.

Also, if there is any piece of pregenerated code included either in the repository or in the tarball I would appreciated:

- If it is documented that there is actually this code.
- If there is described way to regenerate that code.

#9 - 02/07/2023 12:39 PM - hsbt (Hiroshi SHIBATA)

> use of GMP is not documented at all as far as I can say. What is the benefit of enabling it?

Added it at https://github.com/ruby/ruby/commit/31d37e2406d1aa10265944e56807343f91da98fb

#10 - 02/07/2023 04:36 PM - Eregon (Benoit Daloze)

vo.x (Vit Ondruch) wrote in #note-8:

> > and https://github.com/rbenv/ruby-build/wiki#suggested-build-environment
>
> I really wonder what is the source of this information? Just looking at Fedora, I don't understand why patch should be recommended there. And for CentOS / RHEL, why gcc-6? Also RHEL9 has more recent toolchain then RHEL6, so the instructions should be probably differentiated. I don't think that ncurses-devel is needed anymore.

It is what people have found to be necessary.
patch is needed by ruby-build itself because it needs it to apply patches and OpenSSL 1.1.1q was pretty broken, they didn't even check their own CI before release :/ and didn't ship a fixed release for a long time:
https://github.com/rbenv/ruby-build/blob/a6976a5af32443ae0af41fa7713c6b1e8d629ed3/bin/ruby-build#L1152 (probably could be removed now, anyway, out of scope of this issue).
Re gcc-6 yeah that doesn't sound ideal, does just gcc work on CentOS/RHEL? Any package with a version number in the name is a smell, but some package managers don't offer any alternative.

#11 - 02/07/2023 06:42 PM - sorah (Sorah Fukumori)

I'd support adding a good explanation to cover topics mentioned at https://bugs.ruby-lang.org/issues/19421#note-8

And regarding bundled gems and default gems, we should also have a document to cover what are those and how to exclude from installation, for those who need to distribute as a individual package.

If we're failing to provide useful informations, I think we would be seeing distributors to give up package and we'll loss users (= possible community growth/activity/freshness) or, some people who just want to use Ruby as a dependency, might dislike to see.

#12 - 02/08/2023 05:48 AM - duerst (Martin Dürst)

vo.x (Vit Ondruch) wrote in #note-8:

> hsbt (Hiroshi SHIBATA) wrote in #note-2:
>
>> See https://github.com/ruby/ruby/blob/master/doc/contributing/building_ruby.md
>
> Unfortunately, these documents are kind of useless. They explain what is needed but don't explain why and that is the problem. E.g.
>
> - Why gperf? I am not aware I'd need this for anything creating the tarball via make-snapshot or using the tarball directly.

That's used for creating 'enc/unicode/15.0.0/name2ctype.h' (15.0.0 is different for different Unicode versions).
The generation happens maybe once a year, usually by myself. The generated file is then committed, so usually neither compilation from the tarball nor compilation from github is affected.
[There may be other uses of gperf, but I doubt it.]

> - Are these dependencies supposed to cover also execution of test suite?

Some of the Unicode-related tests aren't executed if the necessary data files are not available. That's done to avoid downloading lots of files from the unicode.org website repeatedly.

> Also, if there is any piece of pregenerated code included either in the repository or in the tarball I would appreciated:
>
> - If it is documented that there is actually this code.
> - If there is described way to regenerate that code.

Yes. In addition to 'enc/unicode/15.0.0/name2ctype.h', the 'enc/unicode/15.0.0/casefold.h' and 'lib/unicode_normalize/tables.rb' are pregenerated files related to internationalization/Unicode.

#13 - 02/08/2023 08:57 AM - nobu (Nobuyoshi Nakada)

duerst (Martin Dürst) wrote in #note-12:

>> - Why gperf? I am not aware I'd need this for anything creating the tarball via make-snapshot or using the tarball directly.
>
> That's used for creating 'enc/unicode/15.0.0/name2ctype.h' (15.0.0 is different for different Unicode versions).
> The generation happens maybe once a year, usually by myself. The generated file is then committed, so usually neither compilation from the tarball nor compilation from github is affected.
> [There may be other uses of gperf, but I doubt it.]

It is used also for lex.c, enc/jis/props.h and a couple of extension libraries.
Anyway these files are all committed.

#14 - 02/10/2023 01:54 AM - mame (Yusuke Endoh)

I think it is good to have documentation that answers package distributors' questions. However, in my experience, such documentation often goes unmaintained if it is put in a place where only the committers can edit.

Just a thought. Since the readers of the document are not all users but only a limited number of package distributors, why not put it in this redmine wiki rather than in the repository? You could add a section like "For Package Distributors" in the document below.

https://bugs.ruby-lang.org/projects/ruby/wiki/Main

I think @ioquatix (Samuel Williams) and @vo.x already has the permissions to edit this wiki. If @segaja wants, I'll give him the permissions.

#15 - 02/11/2023 09:47 AM - rubyFeedback (robert heiler)

I don't think the ruby core team can decide for distributions
what they want to enable. I remember how debian used to remove
mkmf which caused issues for those who wanted to install some
gems that were not available in the debian repositories. They

then thought it was the ruby core team's fault that debian
removed mkmf. None from the core team told debian to do so.

What could be helpful is some kind of published list or text
document that quickly shows which gems are internal and
enabled by default, so that people can follow that in order
to decide what they may want to consider for inclusion into
ruby on a specific distribution. But ultimately it's really
up to a distribution to decide on this. Personally I always
dislike distribution maintainers deciding this for me, so
I go with the explicit ./configure variant (see nobu's patch
e. g. where I tend to use at the least this:
--with-ext=readline,openssl,+).

#16 - 02/12/2023 09:06 AM - ioquatix (Samuel Williams)

@mame (Yusuke Endoh) my experience of the wiki is that it suffers the same fate. I prefer that people submit changes via PRs anyway.

#17 - 02/12/2023 09:27 AM - mame (Yusuke Endoh)

If you really think they are destined for the same fate, the wiki is the way to go. Please do not put something in the repository that will not be maintained.

#18 - 02/12/2023 10:50 AM - Eregon (Benoit Daloze)

I would think the wiki has a much higher chance of getting out of date (from experience that's been the case for how-to-build docs and how-to-contribute docs, the how-to-build in repo is better maintained).
Being in the repository should actually be much better for that.

Anyway, https://github.com/ruby/ruby/blob/master/doc/distribution.md as it is is general information that won't get out of date much.

What has not been said so far is specifics that change over time (e.g. which openssl version is needed) and deep knowledge about dependencies is something that's not documented and likely will not be or not extensively (because that would be much harder to keep up to date).
That's something someone just learns while building and using and learning about Ruby.
It's of course very useful if someone distributing Ruby knows Ruby and its dependencies well.

#19 - 02/12/2023 09:31 PM - hsbt (Hiroshi SHIBATA)

Why don't you use https://wiki.archlinux.org/title/Ruby_package_guidelines for Arch Linux?

And distributors already used wiki for distributing ruby.

- https://fedoraproject.org/wiki/Packaging:Ruby
- https://wiki.debian.org/Teams/Ruby/Packaging
- https://en.opensuse.org/openSUSE:Packaging_Ruby
- https://wiki.gentoo.org/wiki/Ruby
- https://wiki.freebsd.org/Ruby

I still don't understand why we need to maintain for distributors on our repository instead of their documents.

#20 - 02/12/2023 11:40 PM - ioquatix (Samuel Williams)

> I still don't understand why we need to maintain for distributors on our repository instead of their documents.

As I said earlier, nothing is necessary.

However, what I'm seeing, is that there are lots of people asking questions about how to package ruby for operating systems distributions.

"We should listen to the questions being asked and use those questions to drive improvements to documentation."

Is what I already said.

I can also add, that the choices we make as part of Ruby affect downstream distribution and we should document that clearly, e.g. dependencies, gems, standard library gems vs default gems, packaging standards, security handling, etc, all fall within the scope, to make a better Ruby eco-system for everyone.

> If you really think they are destined for the same fate, the wiki is the way to go. Please do not put something in the repository that will not be maintained.

Sorry, I wasn't clear. I was saying "The wiki is almost always out of date and any documentation that ends up there will suffer the fate you mentioned".

I personally think documentation in the ruby git repository has a much higher chance of being kept up to date. I agree as @Eregon (Benoit Daloze) said.

It's also better that such documentation is versioned, as historical changes can matter for older releases, or maintainers working on old releases.

Finally, I don't personally always have internet access, so I appreciate having the documentation in one place. Spreading important documentation into lots of different places is hard to manage.

> @hsbt's list of links

I personally think it could be nice to add these links to the distribution.md as if someone who is new to packaging Ruby is looking for examples, that is a great list of existing guidelines and ideas about packaging Ruby

#21 - 02/13/2023 08:17 AM - sorah (Sorah Fukumori)

while individual distributions' policy and guideline should be kept in their docs and wiki, I am positive to have general guide and recommendation in ruby/ruby to prevent confusion or disparity between distributors. If the central doc helps distributors it'll contribute a lot to the ecosystem.

I agree there is a possibility of neglect, however we can give it a try if we have a few people who would like to maintain, and we should.