Ruby - Feature #17473

Make Pathname to embedded class of Ruby

12/26/2020 12:00 PM - hsbt (Hiroshi SHIBATA)

Status:	Closed	
Priority:	Normal	
Assignee:	akr (Akira Tanaka)	
Target version:		

Description

pathname is one of most useful utility class of Ruby. I'm happy to use Pathname without require it.

Any thought?

Associated revisions

Revision 03800bff - 07/15/2025 10:57 AM - byroot (Jean Boussier)

Make Pathname#mkpath builtin

[Feature #17473]

History

#1 - 12/26/2020 12:45 PM - Eregon (Benoit Daloze)

If we do so, could we actually define most of Pathname in Ruby, and not in C?

Right now, https://github.com/ruby/ruby/blob/3fc53de5c961cc8fa2b6acbd63874b89fe709520/ext/pathname/pathname.c is essentially just a bunch of rb_funcall() calls, which are no faster than Ruby code, but makes it harder to reuse on other implementations, and less readable and harder to maintain.

#2 - 12/27/2020 10:57 AM - mame (Yusuke Endoh)

- Assignee set to akr (Akira Tanaka)

The proposal lacks one background, so I'd like to add it. Rubygems cannot allow uesrs to choose the version of a gem that rubygems itself are using. So, we want to make Rubygems independent with any gems. According to @hsbt (Hiroshi SHIBATA), Rubygems is using Pathname, FileUtils, Tsort, etc. Though FileUtils and Tsort are relatively easy to be removed from the dependencies, Pathname looks difficult.

In addition, of course, Pathname is widely used, and the API of Pathname looks stable. So I also think it deserves built-in.

#3 - 12/27/2020 04:16 PM - Eregon (Benoit Daloze)

I forgot to mention, +1 from me for Pathname in core, if it's written mostly in Ruby.

#4 - 12/27/2020 07:19 PM - Dan0042 (Daniel DeLorme)

mame (Yusuke Endoh) wrote in #note-2:

> Though FileUtils and Tsort are relatively easy to be removed from the dependencies, Pathname looks difficult.

Isn't Pathname dependent on FileUtils though?

#5 - 12/28/2020 12:30 AM - mame (Yusuke Endoh)

Dan0042 (Daniel DeLorme) wrote in #note-4:

> Isn't Pathname dependent on FileUtils though?

Yes. It uses only FileUtils.mkpath and rm_r. Rubygems also uses them, so @hsbt (Hiroshi SHIBATA) is preparing to propose making a very limited set of FileUtils methods built-in, too.

#6 - 12/28/2020 12:38 PM - Eregon (Benoit Daloze)

I thought about FileUtils too, but it's required lazily for these two methods.
So I think it might be fine to move only Pathname to core, and accept that Pathname#{mkpath,rmtree} requires fileutils when used.

I think having part of FileUtils defined in core would be confusing.

#7 - 12/30/2020 03:05 PM - shyouhei (Shyouhei Urabe)

JFYI pathname was born as a pure-ruby library, then was eventually translated into C (in 4bf3cb5ba9c0242bd5a6d0d55b7db9f837c09edf). Don't know the reason behind that move though. @akr (Akira Tanaka) do you remember?

#8 - 12/30/2020 03:10 PM - Eregon (Benoit Daloze)

shyouhei (Shyouhei Urabe) wrote in #note-7:

> JFYI pathname was born as a pure-ruby library, then was eventually translated into C (in 4bf3cb5ba9c0242bd5a6d0d55b7db9f837c09edf). Don't know the reason behind that move though. @akr (Akira Tanaka) do you remember?

I know, I'd like to undo that change (I can make a PR), except for the parts which are significantly faster in C (I think only Pathname#<=> / path_cmp).
Then, we could share more of Pathname between Ruby implementations, and avoid maintaining both a Ruby and C version.
There is a copy in TruffleRuby, in JRuby and in Rubinius right now due to that commit :/
@akr (Akira Tanaka) Would that be OK?

#9 - 12/31/2020 12:01 AM - hsbt (Hiroshi SHIBATA)

> Then, we could share more of Pathname between Ruby implementations, and avoid maintaining both a Ruby and C version.

I'm not sure how embedded pure-Ruby implementation to core classes. The above request is the different request.

#10 - 12/31/2020 09:22 AM - duerst (Martin Dürst)

hsbt (Hiroshi SHIBATA) wrote in #note-9:

> I'm not sure how embedded pure-Ruby implementation to core classes. The above request is the different request.

I support making Pathname part of core Ruby. It's an extremely convenient library that I use very often.

Just for the record, String#unicode_normalize is part of a core class but is implemented in pure Ruby. It's a single case, and includes loading code on demand. If we do more pure-Ruby implementations for core classes, we should think again about the best way to do it.

#11 - 01/01/2021 11:53 AM - Eregon (Benoit Daloze)

hsbt (Hiroshi SHIBATA) wrote in #note-9:

> I'm not sure how embedded pure-Ruby implementation to core classes. The above request is the different request.

It's also related, if the decision was to move all of Pathname to C (due to some reason of moving it to core), I would be strongly against it.
I think concretely we can just have pathname.rb at the root, similar to e.g. dir.rb.

#12 - 01/12/2021 04:43 AM - akr (Akira Tanaka)

shyouhei (Shyouhei Urabe) wrote in #note-7:

> JFYI pathname was born as a pure-ruby library, then was eventually translated into C (in 4bf3cb5ba9c0242bd5a6d0d55b7db9f837c09edf). Don't know the reason behind that move though. @akr (Akira Tanaka) do you remember?

I had a plan that Pathname contains a FD optionally as root directory and
use openat() and other *at system call.
No progress after translation to C, though

#13 - 01/06/2022 08:12 PM - Eregon (Benoit Daloze)

I plan to make a PR to move Pathname back to Ruby for all methods which don't significantly gain from being written in C.
I'm coming from https://github.com/oracle/truffleruby/issues/2559, which is additional motivation for it, the current situation hurts compatibility between Ruby impls for seemingly no gain.

We can still decide to keep pathname as stdlib or move it to core either way (there is already ext/pathname/lib/pathname.rb anyway, it'd just become bigger and ext/pathname/pathname.c smaller).

#14 - 01/06/2022 08:17 PM - Eregon (Benoit Daloze)

Regarding openat(), 4bf3cb5ba9c0242bd5a6d0d55b7db9f837c09edf was 11 years ago, so I assume there is little demand or need.

It's easy enough to just use absolute paths before chdir (or avoid using them after chdir) or keep the old CWD as a Pathname instance, and of course some platforms don't have openat().

#15 - 01/07/2022 09:25 AM - deivid (David Rodríguez)

For what it's worth, although for unrelated reasons, I completely rewrote Pathname in pure Ruby here: https://github.com/rubygems/rubygems/pull/4992.

#16 - 04/03/2024 03:50 AM - hsbt (Hiroshi SHIBATA)

- Status changed from Open to Assigned

#17 - 06/23/2025 08:10 PM - Eregon (Benoit Daloze)

I (finally) made a PR to define most of Pathname in Ruby: https://github.com/ruby/pathname/pull/53

#18 - 07/10/2025 05:16 AM - matz (Yukihiro Matsumoto)

Regarding the original proposal, I'd accept making Pathname built-in, considering its wide accepting among the community, especially with Rails. Whether to keep the built-in Pathname in the C implementation or implement it in Ruby is left for future discussion.

Matz.

#19 - 07/15/2025 08:30 AM - hsbt (Hiroshi SHIBATA)

- Status changed from Assigned to Closed

I made Pathname to embedded core class now.

- https://github.com/ruby/ruby/pull/13873
- https://github.com/ruby/ruby/pull/13885
- https://github.com/ruby/ruby/pull/13887

Pathname#find, Pathname#mkpath, Pathname#rmtree and Pathname.mktmpdir is remained on lib/pathname.rb. So, the users need to require pathname.rb for that methods. We should migrate them to pathname_builtin.rb or pathname.c without extra dependencies.

#20 - 07/15/2025 10:23 AM - deivid (David Rodríguez)

Would it make sense to make find, fileutils, and tmpdir also builtin, given that pathname depends on them?

#21 - 07/15/2025 07:06 PM - Eregon (Benoit Daloze)

Discussion at
https://github.com/ruby/dev-meeting-log/blob/master/2025/DevMeeting-2025-07-10.md#feature-17473-make-pathname-to-embedded-class-of-ruby-eregon

I am disappointed, https://github.com/ruby/pathname/pull/53 makes the code clearly more maintainable and a lot easier to read, there is no value to have it in C.
I don't see much value to have Pathname as a core class, require "pathname" is no big deal and since it depends on other gems & stdlib it's awkward.
Having a partially-defined Pathname in core (missing some methods) seems the worst outcome.

As pathname is already a gem, it also means we'll have two unsynchronized copies of the code, and with enough time maybe subtle behavior differences, this seems something nobody wants.

@matz (Yukihiro Matsumoto) this decision is adding a lot of work to CRuby, and other Ruby implementations.
Pathname or most of it should be defined in Ruby, having multiple copies of it in C, Java, Ruby, etc means useless duplication and things will get out of sync needlessly.

> @nobu (Nobuyoshi Nakada): if Pathname is embeded to the core, I prefer the C implementation to Ruby impl.

Why? I see only disadvantages to that.

> @ioquatix (Samuel Williams) for the purpose of JIT isn't it better to write more Ruby in Ruby?
> @shyouhei (Shyouhei Urabe) nobody wants to JIT a path manipulation.

https://github.com/ruby/pathname/pull/53#issuecomment-2989134661 shows the Ruby code is faster than the C extension for Pathname.
Even from a performance point of view it's bad to implement Pathname methods in C (for most methods).

#22 - 07/15/2025 07:27 PM - Eregon (Benoit Daloze)

I have merged https://github.com/ruby/pathname/pull/53, it is clearly better for the maintenance of the gem at least.
Now we need to figure out if we want to keep the copy in core in sync with the gem or not (if not it's going to be a "fun" gotcha that the pathname gem is faster than core Pathname).

#23 - 07/16/2025 02:10 AM - Dan0042 (Daniel DeLorme)

- akr: To use the following four methods, require "pathname" is needed for a while
  - find, mkpath, rmtree, mktmpdir
  - rationale: they require other libraries (find gem, fileutils, tmpdir)
  - in future, they (except mktmpdir) should be rewritten in pure C (or rbinc) without dependency to find and fileutils

I don't understand this at all. What's the problem with having require "fileutils" inside #mkpath ? Those dependencies are part of the stdlib anyway, so it's not like they could be missing. Is it for some reason forbidden to call require in a method of a core class?

#24 - 07/16/2025 05:27 AM - hsbt (Hiroshi SHIBATA)

> I have merged https://github.com/ruby/pathname/pull/53, it is clearly better for the maintenance of the gem at least.

I reverted that. Please don't merge without consensus. At least, no one strongly against migrate C to Ruby in Dev Meetings.

The transition from C to Ruby should be done gradually, in reviewable chunks.

#25 - 07/16/2025 07:24 AM - Eregon (Benoit Daloze)

Dan0042 (Daniel DeLorme) wrote in #note-23:

> I don't understand this at all. What's the problem with having require "fileutils" inside #mkpath ? Those dependencies are part of the stdlib anyway, so it's not like they could be missing. Is it for some reason forbidden to call require in a method of a core class?

I had the exact same thought, this seems the obvious solution.

#26 - 07/16/2025 07:45 AM - Eregon (Benoit Daloze)

hsbt (Hiroshi SHIBATA) wrote in #note-24:

> > I have merged https://github.com/ruby/pathname/pull/53, it is clearly better for the maintenance of the gem at least.
>
> I reverted that. Please don't merge without consensus. At least, no one strongly against migrate C to Ruby in Dev Meetings.
>
> The transition from C to Ruby should be done gradually, in reviewable chunks.

That PR is actually easy to review as explained in its description, it restores pathname.rb as it used to be and includes all changes to pathname.rb since then, see https://github.com/ruby/pathname/pull/53/files/3736eab91f783cb087f873b99c25ffb7633ed041.
IOW, it only adds code in pathname.rb and removes code in pathname.c, and then has separate commits for all other changes.
Making multiple smaller PRs does not make sense, especially since the Ruby code is just the one from before the migration to C.

I remade the PR as https://github.com/ruby/pathname/pull/57, please review it and share your concerns there.

What is the plan for the code in the gem and the code in core?
Will they be sync'd, how?
From what I have seen changes in core have been done seemingly without caring about the gem, so it seemed the intention was to not sync but effectively fork?
We also cannot make the gem empty in general as the gem needs to keep working on Ruby 3.4 and older at least.

#27 - 07/18/2025 06:12 AM - hsbt (Hiroshi SHIBATA)

Eregon (Benoit Daloze) wrote in #note-26:

> That PR is actually easy to review as explained in its description, it restores pathname.rb as it used to be and includes all changes to pathname.rb since then, see https://github.com/ruby/pathname/pull/53/files/3736eab91f783cb087f873b99c25ffb7633ed041.
> IOW, it only adds code in pathname.rb and removes code in pathname.c, and then has separate commits for all other changes.
> Making multiple smaller PRs does not make sense, especially since the Ruby code is just the one from before the migration to C.

Please separate the small PRs. I want to reduce the side effect like https://github.com/ruby/ruby/pull/13906.

And if you have another issue or proposal, please file new issue.

#28 - 07/18/2025 08:47 AM - Eregon (Benoit Daloze)

hsbt (Hiroshi SHIBATA) wrote in #note-27:

> Please separate the small PRs. I want to reduce the side effect like https://github.com/ruby/ruby/pull/13906.

Let's discuss your concerns about the PR on https://github.com/ruby/pathname/pull/57 directly.