Ruby master - Misc #14274

Merge Std-Lib Time Class into Core

01/02/2018 11:27 PM - ecbrodie (Evan Brodie)

Status:	Open	
Priority:	Normal	
Assignee:		

Description

Please excuse me if this is a duplicate or already asked-and-answered. I won't be offended by an automatic close.

I personally find it confusing when to use the Core version of the Time class versus the Std-Lib require "time" version of Time. I fail to see what makes the Std-Lib version of that class so differentiable from Core to necessitate an explicit require statement. Merging these two into one Core version would cut some amount of complexity in using the Ruby language, however small a complexity it may be, to improve user experience by automatically having all methods in Time available.

As an example of this complexity in action, I was recently working on a Rails app stack today alongside another experienced Ruby developer like myself. We switched over from working on code in the Rails app to working on code in a supporting nested gem. This gem does not have a dependency on Rails, so it does not pull in the monkey-patched Rails version of Time. Thus, when we wanted to write a function that uses Time.parse, we ran into an issue with the Ruby interpreter not recognizing this method. We were initially confused about how this could be happening, for a moment positing that this parse method is unique to Rails. Eventually, we realized that we needed to add require "time" to the codebase, although it took a bit of Googling to realize that there were two different versions of the Time class in the Ruby language. Even though this complexity was small and the fix trivial, we lost 5-10 minutes of productivity to it. We are experienced developers, so imagine the pain that a newer developer with little knowledge of the difference between Core and Std-Lib would suffer.

Hopefully you can see the reasoning that I am applying to this request. I am happy to divulge deeper into the problem if necessary. If you agree, then hopefully this can make it into the next minor version of Ruby. Thank you. :D

Related issues:

Related to Ruby master - Misc #13072: Current state of date standard library	**Open**

History

#1 - 01/03/2018 01:19 AM - shevegen (Robert A. Heiler)

I do not use rails, but I somewhat agree with you in the general sense. I never remember the difference between date and time offhand, for instance.

> I fail to see what makes the Std-Lib version of that class so differentiable from Core to necessitate an explicit require statement.

I do not know either. Perhaps it is legacy, or came from a time where explicit requires where needed more, not sure.

Recently "require 'pp'" became superfluous, which is a nice change (I don't remember who suggested it ... normalperson or I think some other ruby core committer; I also wanted to suggest it but I felt like adding too many issues, so I slowed down suggesting things).

Perhaps on similar grounds or reasons, the same require line for pp - the argument - could also be made for time/date (I mention both since I don't remember the difference offhand).

> Eventually, we realized that we needed to add require "time" to the codebase, although it took a bit of Googling to realize that there were two different versions of the Time class in the Ruby language.

Well, you rails people really should learn ruby ;) - but I agree with you in this context here. If possible, it would be nice if require 'time' would become an integral part, considering that it is quite useful. I use pp much more than time, but in many of my gems, I use time-related

functions. Usually to determine the current day, often
also the current time; sometimes I also have to calculate
like "if input covers 7 days into the future, show which
appointments/exams are upcoming". I guess this is somewhat
common for ruby people, not unlike pp (though I guess even
more people use pp since it pretty-prints arrays and hashes).

By the way, I do not think that it will be auto-closed but
it can take quite some time sometimes to process - I assume
that the core team may want to discuss pros/cons and
usefulness at one of the ~next meetings probably. Matz makes
the final decision but the more people may find something
useful, the higher the chance it may be that it will be
accepted.

> and the fix trivial, we lost 5-10 minutes of productivity
> to it.

Well, I don't think that the 5-10 minutes lost time are
earth-shattering :) but very similar to the reason of
require 'pp', I think a similar reason could be said in
regards to time-functionality. By the way, I am not so
much referring to require 'time' alone but more in general
streamlining and simplifying time-related tasks in ruby.

It does work fine for me, by the way, even without using
any gems, but I somewhat agree with your use case; new
ruby hackers often have to overcome lack of knowledge
and quite some complexity, so this is I think where/why
the proposal is good.

> We are experienced developers, so imagine the pain that
> a newer developer with little knowledge of the difference
> between Core and Std-Lib would suffer.

Perhaps rails developers but not necessarily ruby hackers :D

I am just kidding though, don't take me too seriously. I
agree with the comment - attracting and retaining new people
is also important, when the old hackers get older.

> If you agree, then hopefully this can make it into the
> next minor version of Ruby.

You only have to convince matz in the end. :)
Though convincing other ruby core contributors can help.

I am neither of them, just a random ruby hacker :D

Actually, we even have DateTime ... I could not tell you
what it does, or why it has the name of both time and
date ... :P

There is another thing that I may suggest to clean up a
bit which is the various *popen variants, system,
Io-open ... but that is for another issue request.

+1

#2 - 01/03/2018 06:28 AM - jeremyevans0 (Jeremy Evans)

time requires date, so this would either require date be moved from stdlib to core (including Date and DateTime). While time itself only uses Date._parse and Date._strptime, people could be requiring time to use features provided by date.

I suppose Date._parse and Date._strptime could be switched to Time._parse and Time._strptime, and have those in core (and make Date._parse and Date._strptime call them). That would allow Time.parse and Time.strptime to work without requiring date. However, I don't think the benefit of doing that exceeds the cost.

#3 - 01/05/2018 09:21 PM - naruse (Yui NARUSE)

- Related to Misc #13072: Current state of date standard library added

#4 - 02/01/2018 09:25 PM - ecbrodie (Evan Brodie)

Thank you for the replies so far. I do indeed hope that this issue gets attention from the core Ruby maintainers, maybe even Matz himself.

And just to reiterate here (and I think shevegen agrees within his joking around), this issue is not in any way related to the Rails framework. I feel like even if I wasn't accustomed to Rails's monkey-patches, I would still feel strongly about this request.