Ruby master - Feature #16824

Follow RubyGems naming conventions for the stdlib

05/01/2020 03:33 AM - shan (Shannon Skipper)

Status:	Rejected
Priority:	Normal
Assignee:	
Target version:	

Description

It's been really nice that most gems these days follow the RubyGems naming convention, so you know exactly what to require just from seeing the gem name: https://guides.rubygems.org/name-your-gem/

I wonder if it would be possible to add aliases for parts of the stdlib that don't follow the convention for Ruby 3.0. I was thinking maybe shims like lib/optionparser.rb, which just does a require_relative of lib/optparse.rb. The following files are what I'd expect, given the namespaces.

```
new file:    file_utils.rb
new file:    getopt_long.rb
new file:    ip_addr.rb
new file:    open_struct.rb
new file:    open_uri.rb
new file:    option_parser.rb
new file:    p_store.rb
new file:    r_doc.rb
new file:    secure_random.rb
new file:    t_sort.rb
new file:    weak_ref.rb
```

Eventually the old name could be the shim and new one actually contain the code on the path to deprecating old names in some future Ruby.

Anyway, I just wanted to put the suggestion out there to adopt naming conventions for the stdlib as it's gemified. Ruby 3 seems like a nice time. :)

History

#1 - 05/01/2020 05:24 AM - nobu (Nobuyoshi Nakada)

- Status changed from Open to Rejected

#2 - 05/01/2020 01:48 PM - naruse (Yui NARUSE)

Features should have use cases.

#3 - 05/01/2020 02:19 PM - retro (Josef Šimánek)

naruse (Yui NARUSE) I was thinking about the same for a while as well. My use-case in here is simple. I often require 'secure_random' since that should be standard filename for file defying SecureRandom class/module, but actually I end up with error and I realize I need to realize I need to require 'securerandom'. Making 'secure_random' shortcut to require 'securerandom' would be easy enough to maintain (and also original securerandom could be deprecated later). Also I think all docs should be updated to use this "standard" requires.

Doing this kind of "shortcut" is AFAIK common pattern used in various gems as well.

PS: By naming "standard" I mean this RubyGems guide - https://guides.rubygems.org/name-your-gem/.

#4 - 05/01/2020 02:22 PM - retro (Josef Šimánek)

The same also works for English.rb which is part of Ruby itself. It is very confusing to do require 'English' (require file with initial capital letter).

#5 - 05/01/2020 02:54 PM - zverok (Victor Shepelev)

naruse (Yui NARUSE)nobu (Nobuyoshi Nakada) The "use case" for this is Rubyists nowadays have a very strong intuition that "Foo::BarBaz" module typically required via foo/bar_baz path. This convention is encouraged by tutorials (including RubyGems' linked above, with RubyGems being part of Ruby!) and linters.

I believe it is desirable that Ruby's core and standard library API provided an exemplary following the community conventions and intuitions, where possible.

For my 15 years with Ruby, for the first 8 or 9 I've constantly struggled with require "file_utils" and require "open_struct" → oh, I remember there was a quirk require "openstruct" → ugh, oh, require "ostruct".

The change proposed is of negligible size (just creating 8 1-line files), will create zero problems and maintenance burden, and will save a lot of people from everyday frustration.

(I am saying 8 because I believe RDoc, PStore, and TSort shouldn't be affected -- it is kind of abbreviations, RubyGems tutorial even have example of rdoc-data containing RDoc::Data)

#6 - 05/01/2020 07:33 PM - marcandre (Marc-Andre Lafortune)

We can't do this in general, sadly. For example, there exist two gems file-utils and file_utils, both of which use require 'file_utils' to be used.

Some other gems are conflict free (e.g. neither open_struct or open-struct are reserved) so it could very well be done without any downside.

Would it not be a good idea to at least reserve gem names like open_struct or open-struct?

#7 - 05/01/2020 07:33 PM - marcandre (Marc-Andre Lafortune)

- Status changed from Rejected to Open

#8 - 05/01/2020 11:53 PM - retro (Josef Šimánek)

marcandre (Marc-Andre Lafortune) AFAIK some gem ownerships were already transferred back to claim std-lib names. Looking at those examples, majority of those conflicts could be solved in the same way if needed (long-term).

The main question in here is if Ruby would like to be consistent in this and how strict this consistency should be required. I agree on rdoc example regarding shortcuts and in general I prefer as well to store module DSL definition into dsl.rb file instead of d_s_l.rb. From the other point of view, each exception (inconsistency like this) makes this whole unification less useful.

#9 - 05/02/2020 01:42 AM - shevegen (Robert A. Heiler)

The optionparser situation is indeed quite ... peculiar to me:

```
require 'optparse'
OptionParser.new
```

My brain has a hard time with it. I'd prefer:

```
require 'optionparser'
```

or

```
require 'option_parser' # <- would probably be best.
```

Anyway - I don't have a strong preference, but personally I agree with havenwood (shan).

If one issue is backwards compatibility or other gems that do the same, then that
could go for consideration in ruby 4.0 perhaps, so years from now on.

zverok wrote:

> I remember there was a quirk require "openstruct" → ugh, oh, require "ostruct".

Yeah, similar to optparse. I guess you can reason that 'ostruct' is shorter to
write than 'openstruct', but I myself prefer a consistent naming. But as said,
I don't have a big preference really.

#10 - 05/02/2020 02:47 AM - nobu (Nobuyoshi Nakada)

shevegen (Robert A. Heiler) wrote in #note-9:

> The optionparser situation is indeed quite ... peculiar to me:
>
> ```
> require 'optparse'
> OptionParser.new
> ```
>
> My brain has a hard time with it. I'd prefer:
>
> ```
> require 'optionparser'
> ```

It has been available since 6 years ago, for your [Feature #9864].

```
$ ruby -roptionparser -e 'ARGV.parse!' -- --help
Usage: -e [options]
```

#11 - 05/22/2020 01:33 PM - hsbt (Hiroshi SHIBATA)

- Status changed from Open to Rejected

The first proposal was rejected. We should discuss each stdlib like ostruct.

#12 - 05/26/2020 01:49 AM - marcandre (Marc-Andre Lafortune)

- Status changed from Rejected to Open

Let's not close this because the original request is not be accepted at 100%

Any volunteer to double-check the list and figure out which gem names are available and which could possibly be reclaimed?

I can help to register the names and make the aliases.

#13 - 05/26/2020 02:03 AM - nobu (Nobuyoshi Nakada)

- Status changed from Open to Rejected

Nonsense.

#14 - 05/26/2020 02:20 AM - yuki24 (Yuki Nishijima)

While I understand the intention of this request, I'm not sure this would be practically useful for us. As mentioned above, there are definitely gem names that people are already too familiar with, that we probably don't want to rename, and shifting the direction at this point would create even more confusion later on (e.g. why can't I require secure_random in Ruby 2.7?). For some of the gems like json and rdoc, it simply wouldn't make sense. And even if all the standard/bundled gems were to follow the naming conventions, the entire world would never be as consistent as you wish because of the existence of popular gems such as activesupport, rest-client, etc.

Also, see how much of effort the rest-client maintainers have put into this: https://github.com/rest-client/rest-client/tree/master/lib. If we were to do the same, we could easily fall down the rabbit hole of claiming a lot more names than initially anticipated.

With all that said, I understand how frustrating it could be when you can't remember the gem name you want to require. In response to that, I have added a "Did you mean?" feature to LoadError to help resolves issues like this quickly: https://github.com/ruby/did_you_mean/pull/143

```
require "open_struct"
# LoadError (cannot load such file -- open_struct)
#   Did you mean?  ostruct

require "file_utils"
# LoadError (cannot load such file -- file_utils)
#   Did you mean?  fileutils
```

This way you would get gem name suggestions on an LoadError and it should save you from not remembering thae gem name correctly. Going forward, other static analysis tools like sorbet and steep should also be able to do the same statically rather than doing so at runtime.

#15 - 05/26/2020 02:59 AM - marcandre (Marc-Andre Lafortune)

nobu (Nobuyoshi Nakada) wrote in #note-13:

> Nonsense.

Nobu, I feel deeply offended by your impoliteness and lack of respect.

yuki24 (Yuki Nishijima) wrote in #note-14:

```
require "open_struct"
# LoadError (cannot load such file -- open_struct)
#   Did you mean?  ostruct

require "file_utils"
# LoadError (cannot load such file -- file_utils)
#   Did you mean?  fileutils
```

This is great, well done!