The next dev meeting

Date: 2020/11/20 13:00-17:00
Place/Sign-up/Agenda/Log: [https://github.com/ruby/dev-meeting-log/blob/master/DevelopersMeeting20201120Japan.md](https://github.com/ruby/dev-meeting-log/blob/master/DevelopersMeeting20201120Japan.md)

- Dev meeting *IS NOT* a decision-making place. All decisions should be made on the bug tracker.
- Dev meeting is a place we can ask Matz, nobu, nurse, and other developers directly.
- Matz is a very busy person. Take this opportunity to ask him. If you cannot attend, a different attendee can ask instead of you (if the attendee can understand your issue).
- We will write a log about the discussion in a file or on each ticket in English.
- All activities are best-effort (keep in mind that most of us are volunteer developers).
- The date, time, and place are scheduled according to when/where we can reserve Matz’ time.
- *DO NOT* discuss them on this ticket, please.

Call for agenda items

If you have a ticket that you want Matz and other committers to discuss, please post it on this ticket in the following format:

- [Ticket ref] Ticket title (your name)
  * Comment (A summary of the ticket, why you put this ticket here, what points should be discussed, etc.)

Example:

- [Feature #14609] `Kernel#p` without args shows the receiver (ko1)
  * I feel this feature is very useful and some people say :+1:, so let's discuss this feature.

- Comment deadline: 2020/11/13 (one week before the meeting)
- The format is strict. We'll use this script to automatically create a markdown-style agenda. We may ignore comments that do not follow the format.
- Your comment is mandatory. We cannot read the entire discussion of the ticket within a limited time.

Related issues:

Related to Ruby master - Misc #14770: [META] DevelopersMeeting Open

History

#1 - 10/31/2020 12:55 AM - mame (Yusuke Endoh)
- Related to Misc #14770: [META] DevelopersMeeting added

#2 - 10/31/2020 01:53 AM - sawa (Tsuyoshi Sawada)
- Description updated

#3 - 11/02/2020 11:15 AM - hsbt (Hiroshi SHIBATA)

- [Feature #17303] Make webrick to bundled gems or remove from stdlib
  - Any objection?

#4 - 11/02/2020 08:45 PM - jeremyevans0 (Jeremy Evans)

- [Feature #17143] Improve support for warning categories (jeremyevans0)
  - I researched the Python warning categories, and provided some analysis.
  - Can we decide on warning categories so they can be implemented in time for Ruby 3?
Bug #10845 Subclassing String (jeremyevans0)
- In #6087, matz decided that in Ruby 3.0, methods for Array subclasses should return arrays and not subclass instances.
- The same issues that apply to Array subclasses also apply to String subclasses.
- Do we want to make String subclass methods return strings instead of subclass instances?
- If so, is the pull request OK?
- Note that changing the behavior of String methods will require changes in Rails.

Bug #11022 opening an eigenclass does not change the class variable definition context (jeremyevans0)
- This was discussed in the December 2019 meeting, but a decision was not made.
- Considering #14012 was rejected last meeting, can we also reject this?

Bug #7844 include/prepend satisfiable module dependencies are not satisfied (jeremyevans0)
- This was discussed in the December 2019 meeting, but a decision was not made.
- Considering the other improvements to include/prepend in 3.0, it would be a good time to fix this.

Bug #11213 defined?(super) ignores respond_to missing? (jeremyevans0)
- I agree with Koichi that it is best to accept the current behavior as spec.
- Can we reject this?

#5 - 11/03/2020 10:13 AM - byroot (Jean Boussier)

Feature #13381 Expose rb_fstring and its family to C extensions (byroot)
- The feature itself was approved, however we're still waiting on an agreement on the naming of one function (https://github.com/ruby/ruby/pull/3586)
- Both json and msgpack now call String#-@, and could benefit from large allocation reduction if they could call rb_fstring_cstr instead.

#6 - 11/03/2020 08:25 PM - marcandre (Marc-Andre Lafortune)

Bug #17197 Yielding arity for Hash methods (marcandre)
- Hash#each with arity 1: confirmed?
- #select, #keep_if, #delete_if, #reject and to_h should also be changed, right?
- #map: should only accept arity 1, right?

#7 - 11/04/2020 12:28 AM - mrkn (Kenta Murata)

Feature #17291 Optimize __send__ call
- Improve the performance of __send__ call by the following ways
  1. Omit emitting __send__ calls for literal method name cases
  2. Introduce a new instruction sendsym to skip __send__ call for dynamic method name cases
- I guess it is no problem to skip __send__ call because the following warning says "redefining __send__ may cause serious problems" for a long time
- I want to know Matz's reaction to this

#8 - 11/04/2020 10:46 AM - Eregon (Benoit Daloze)

Feature #16043 $LOAD_PATH.resolve_feature_path should not raise (erogen)
- Seems good to me, OK to merge?

#9 - 11/05/2020 10:38 AM - Eregon (Benoit Daloze)

Feature #17100 Ractor naming (erogen)
- What should be the method name to send a message? send seems problematic as many have reported. We should decide soon. Many good other names are listed in the issue (notably in https://bugs.ruby-lang.org/issue/17100#note-19 and https://bugs.ruby-lang.org/issue/17100#note-43).
- Specifically, Ractor#send conflicts with Kernel#send, which is a violation of the Liskov substitution principle.
- There has been (FAIKA) no official deprecation of Kernel#send, so it seems inadvisable to use the name for other semantics for Ractor. Deprecating Kernel#send seems a way to require 100 000s of changes in gems for very few little reasons, so it seems unfeasible in practice anyway.
- When should send and __send__ be used? It seems gems prefer to use send (>10x more used than __send__), and only use __send__ when required. IMHO __send__ feels like Python's __add__, which I find unpretty, and feels like C's poor way of namespace things.

#10 - 11/05/2020 11:12 AM - Eregon (Benoit Daloze)

Feature #17307 A way to mark C extensions as thread-safe, Ractor-safe, or unsafe (erogen)
- I'd like feedback on this. Any idea for a good way to mark C extensions?
- How about defining symbols like foo_is_thread_safe/foo_is_ractor_safe to mark C extensions?

#11 - 11/07/2020 02:39 PM - Eregon (Benoit Daloze)

Feature #16786 Light-weight scheduler for improved concurrency (erogen)
- matz (Yukihiro Matsumoto): The Fiber.set_scheduler(value) API doesn't make any sense to me and is very confusing, please see my
comments there (summary: https://bugs.ruby-lang.org/issues/16786#note-70).

- Anything wrong with the obvious and clear Thread.current.scheduler = value or Thread.current.fiber_scheduler = value?

#12 - 11/09/2020 01:28 PM - zverok (Victor Shepelev)

- [Feature #17312] New methods in Enumerable and Enumerator::Lazy: flatten, product, compact (zverok)
- If there would be no objections, I'd prepare a PR

#13 - 11/26/2020 02:13 PM - mame (Yusuke Endoh)

- Status changed from Open to Closed
- Description updated