Status: Closed
Priority: Normal
Assignee:

The next dev meeting

Date: 2020/10/26 13:00-17:00
Place/Sign-up/Agenda/Log: https://github.com/ruby/dev-meeting-log/blob/master/DevelopersMeeting20201026Japan.md

- Dev meeting IS NOT a decision-making place. All decisions should be done at 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 can not attend, other attendees can ask instead of you (if attendees can understand your issue).
- We will write a log about the discussion to a file or to 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's time.
- DO NOT discuss then on this ticket, please.

Call for agenda items

If you have a ticket that you want matz and committers to discuss, please post it into 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 point 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 discuss this feature.

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

History

#1 - 09/28/2020 11:24 PM - mame (Yusuke Endoh)
- Related to Misc #14770: [META] DevelopersMeeting added

#2 - 10/01/2020 01:34 PM - parker (Parker Finch)

- [Feature #17016] Enumerable#scan_left (parker)
  * This is a proposal to add a method to enumerable that accumulates the results of injecting an operation over the enumerable. There is interest in adding this method, but concern about the name. I think that #accumulate is a good name.
  * Should we add this method?
  * Is the name #accumulate acceptable?

#3 - 10/01/2020 07:13 PM - greggzst (Grzegorz Jakubiak)

- [Feature #16851] Ruby hashing algorithm could be improved using Tabulation Hashing
  * Improves hashing algorithm
[Bug #14012] NameError is raised when use class variables in Refinements (jeremyevans0)
- Refinement methods cannot reference class variables of the classes/modules they refine.
- Is this a bug or expected behavior?
- If a bug, how should class variable lookup work in refinement methods?

[Bug #13768] SIGCHLD and Thread dead-lock problem (jeremyevans0)
- You can end up with thread deadlock even when a signal handler could result in breaking the deadlock.
- I don't think this is a bug, otherwise we would have to skip deadlock detection for all custom signal handlers.
- Do we want add a feature such as Thread.ignore_deadlock = true so that we can support this approach?

[Bug #12489] Kernel::Rational raises TypeError though given denominator returns 1 by to_int (jeremyevans0)
- Do we want Kernel#Rational to fallback to checking for to_int if to_r is not defined?
- If so, is the patch acceptable?

[Feature #17143] Improve support for warning categories (jeremyevans0)
- Based on feedback at last month's developer meeting, I'm now proposing adding only two new categories: :uninitialized_ivar and :redefine.
- I have a pull request that adds support for those two categories and adds the :deprecated category to additional warnings, it is OK?

[Feature #17187] Add connect_timeout to TCPSocket (glass)
- Could I merge it?

[Feature #16746] Endless method definition
- The following was discussed at DevelopersMeeting20200514Japan

Conclusion:
- matz: Let me have some time to consider this issue
  - def foo() = or def foo() :
  - def foo(n) = n => @x
  - def foo=bar may break compatibility (def foo= bar; bar; end); So, no-arugment mark () is mandatory. But I may reconsider

[Feature #15921] R-assign (rightward-assignment) operator
- The following was discussed at DevelopersMeeting20200514Japan

Conclusion:
- matz: The motivation is still valid even after pipeline operator is gave up. It is still useful for a long method chain.
- matz: I think an experiment with commit is needed and works. I will explain.

[Feature #17260] Promote pattern matching to official feature
- I'd like to remove single line pattern matching and experimental warnings.

[Feature #17259] Kernel#warn should ignore <internal: entries (eregon)
- https://github.com/ruby/ruby/pull/3647 fixes it, OK to merge?
- Also, RubyGems redefines Kernel#warn for skipping RubyGems' require. That redefinition caused various incompatibilities. If we ignore <internal:, no need to redefine Kernel#warn (I'll make a PR to RubyGems).

[Feature #17269] Promote pattern matching to official feature
- I'd like to remove single line pattern matching and experimental warnings.

[Feature #17259] Kernel#warn should ignore <internal: entries (eregon)
- https://github.com/ruby/ruby/pull/3647 fixes it, OK to merge?
- Also, RubyGems redefines Kernel#warn for skipping RubyGems' require. That redefinition caused various incompatibilities. If we ignore <internal:, no need to redefine Kernel#warn (I'll make a PR to RubyGems).
• [Feature #17171] Have private affect constants and singleton methods definitions too (marcandre)
  ◦ Privacy is important for gems in particular. private_constant is verbose to use.
  ◦ The constants and methods affected would have special flag "legacy public"
  ◦ Ruby 3.0: warn (verbose only) on first access to "legacy public"
  ◦ Ruby 3.1: warn (always) on first access to "legacy public"
  ◦ Ruby 3.2: "legacy public" becomes private

#11 - 10/18/2020 12:36 AM - tenderlovingmaking (Aaron Patterson)

• [Feature #17176] GC.auto_compact / GC.auto_compact=(flag) (tenderlove)
  ◦ Currently compaction is a manual process, eventually I want to make it automatic
  ◦ This feature lets users "opt in" to automatic compaction so we can test extensions
  ◦ Reference updating logic remains the same so "heap integrity" should be the same

#12 - 10/19/2020 12:16 AM - marcandre (Marc-Andre Lafortune)

• [Feature #17265] Have common ancestor Bool for TrueClass and FalseClass (marcandre)
  ◦ Useful for RBS
  ◦ Should have very limited incompatibility

#13 - 10/20/2020 12:04 AM - hsbt (Hiroshi SHIBATA)

• [Feature #17267] Remove Win32API at Ruby 3.0
  ◦ Does anyone oppose it?

#14 - 10/21/2020 04:37 PM - greggzst (Grzegorz Jakubiak)

• [Feature #17277] Make Enumerator#with_index yield row and col indices for Matrix
  ◦ Access Matrix indices in enumerable methods

#15 - 10/22/2020 12:58 AM - ko1 (Koichi Sasada)

• [Bug #17146] Queue operations are allowed after it is frozen
  ◦ It is interesting question what "freeze" mean.
• [Feature #17256] Freeze all Regexp objects
  ◦ I feel it is too heavy to freeze all regexp...
• [Feature #17269] Frozen Process::Status
  ◦ low priority
• [Feature #17145] Ractor-aware Object#deep_freeze
  ◦ Can we define the meaning of class/module?
• [Feature #17279] ObjectSpace.each_object should be restricted on multi-Ractors
  ◦ already introduced, but need confirmation
• [Bug #17268] special global variables which can be accessed from ractors
  ◦ already introduced, but need confirmation
• [Feature #17261] Software transactional memory (STM) for Threads and Ractors
  ◦ Can I merge it with current naming and semantics?
• [Feature #17274] Ractor.make_shareable(obj)
  ◦ already introduced for convenience, but we can change.
• [Feature #17273] shareable_constant_value pragma
  ◦ is it acceptable?

#16 - 10/22/2020 05:17 PM - Eregon (Benoit Daloze)

• [Bug #17280] Dir.glob with FNM_DOTMATCH matches "." and "." and results in duplicated entries (eregon)
  ◦ What's the intended behavior?

#17 - 10/23/2020 05:42 AM - nobu (Nobuyoshi Nakada)

• [Feature #17281] Remove support for mathn
  ◦ mathn has been deprecated.

#18 - 10/24/2020 03:38 AM - koic (Koichi ITO)

• [Ticket #17208] Add Set#compact and Set#compact! methods (koic)
Is this proposal acceptable?

#19 - 10/31/2020 12:54 AM - mame (Yusuke Endoh)
- Description updated

#20 - 10/31/2020 12:55 AM - mame (Yusuke Endoh)
- Status changed from Open to Closed