Please comment on your favorite ticket numbers you want to ask to discuss with your SHORT comment or summary.
(your summary/comment will help us because we don't need to read all of the ticket comments)

DO NOT discuss then on this ticket, please.

Date: 2019/11/28 13:00-17:00
Place, Sign-up, and Log: https://docs.google.com/document/d/1AZ74HXEedKksJwhEUPdInRxUqgchndZPZYAKjGPMsFl/edit#

NOTES

- 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.

Agenda

Next dev-meeting

About 2.7 timeframe

Check security tickets

Discussion

Please comment on your favorite ticket we need to discuss with the following format.

* [Ticket ref] Ticket title (your name)
* your comment why you want to put this ticket here if you want to add.

Your comment is very important if you are no attendee because we can not ask why you want to discuss it.

Example:

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

We don't guarantee to put tickets in the agenda if the comment violates the format (because it is hard to copy&paste).

A short summary of a ticket is strongly recommended. We cannot read all discussion of the ticket in a limited time.

A proposal is often changed during the discussion, so it is very helpful to summarize the latest/current proposal, post it as a comment in the ticket, and write a link to the comment.

Related issues:

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

Open

History

11/04/2021
Carry over:

- [Bug #15912] Allow some reentrancy during TracePoint events (alanwu)
  - I have seen quite a few people confused about Byebug’s REPL not working as they would expect due to this issue. I think it's important to come up with some solution for this in two seven.

- [Feature #16142] Implement code_range in Proc and Method - Ruby master - Ruby Issue Tracking System (osyo)
  - Propose an API to get code position of Proc and Method so that we can get body of them (especially of a Proc).
  - I want to discuss method names and return values

#3 - 10/18/2019 04:12 PM - jeremyevans0 (Jeremy Evans)

- [Feature #16129] Call initialize_clone with freeze: false if clone called with freeze: false (jeremyevans0)
  - Without this, use of clone(freeze: false) on objects not expecting it leads to unexpected state.
  - Allows fixing problems in delegate and set.
- [Bug #15432] Refinements method call to failed (jeremyevans0)
  - OK to fix modules that are refined and prepended by creating an iclass for the module (and not just the module's origin)?
- [Bug #13446] refinements with prepend for module has strange behavior (jeremyevans0)
  - OK to fix case where prepend is used on a refined module after the module is included by creating an origin iclass during inclusion?

#4 - 10/22/2019 10:58 PM - duerst (Martin Dürst)

- [Feature #16262] Enumerable#each_tuple, and [Feature #4539] Array#zip_with
  - zip_with really comes in handy on occasion, and is available in many programming languages, in particular functional programming languages.

#5 - 10/26/2019 12:42 PM - zverok (Victor Shepelev)

(Sorry for being over-active. On the bright side, it could be a final bunch from me)

Carry-over from tickets with feedback:

- [Feature #16122] Struct::Value: simple immutable value object. First version of the proposal was not clear enough, I am sorry for it. Clarified the description, added links and answered some possible questions.
- [Feature #15822] Add Hash#except. Matz: “We didn't see the need for Hash#except yet. Any (real world use-case)? I don't think the name except is the best name for the behavior.” Added real-life examples and name justifications.

This two proposals would make sense only if .. would not be reverted (which, to the best of my understanding, is now doubtful)

- [Feature #16261] Enumerable#each_splat and Enumerator#splat allowing to mitigate 1-element/multi-element enumerators
- [Feature #16264] Real "callable instance method" object. ..method to be a first-class thing, instead of Symbol#to_proc trick

#6 - 11/04/2019 09:13 PM - jonathanhefner (Jonathan Hefner)

- [Feature #15923] Enumerable#filter_map
  - This feature has already been merged, but there may have been some confusion regarding the implementation.
  - It currently selects only truthy values, but should it select all non-nil values, like Array#compact? (See https://bugs.ruby-lang.org/issues/15323#note-17)
  - If so, is the name filter_map still a good choice? Or should it be changed to something like compact_map?

#7 - 11/05/2019 12:20 PM - osyo (manga osyo)

- [Feature #16293] Numbered parameter confirmation in Ruby 2.7
  - Numbered parameter confirmation in Ruby 2.7

#8 - 11/10/2019 02:00 PM - methodmissing (Lourens Naudé)

- [Misc #16291] Introduce support for resize in rb_ary_freeze and prefer internal use of rb_ary_freeze and rb_str_freeze for String and Array types (lourens)
- Builds onto the capacity shrinking feature introduced by rb_str_freeze, targeting Array
- Many internal uses that froze String types did not use the rb_str_freeze variation and could not benefit from resizing capacity on freeze
- Implemented the same for Array
- Let Array#freeze call rb_ary_freeze to expose the shrinking capability to user code too (as recommended by Shyouhei) for parity with String#freeze already doing the same

#9 - 11/14/2019 03:46 AM - alanwu (Alan Wu)
- [Bug #15620] Block argument usage affects lambda semantic
  - I find the current behaviour unreasonable confusing and would like to see improvement, even though the bug doesn't really show up in the real world often.
  - I have a pull request which restores the behaviour found in Ruby 2.4.x and warns about it, based off of matz's comment in [ruby-core:94054].
  - Could someone confirm that is the desired fix? If it is the fix we want, could someone review the PR?

#10 - 11/24/2019 02:25 AM - mame (Yusuke Endoh)
- [Feature #16364] Top-level ruby2_keywords (mame)
  - Currently, there is no top-level ruby2_keywords, which would be unuseful for some simple cases.

#11 - 11/24/2019 03:14 AM - yuki24 (Yuki Nishijima)
- [Feature #16363] Promote did_you_mean to default gem (yuki24)
  - Currently there are two problems with the gem being a bundled gem, one with the availability of the lib and the other about bundler.

#12 - 11/24/2019 09:19 AM - ktsj (Kazuki Tsujimoto)
- [Feature #16355] Raise NoMatchingPatternError when expr in pat doesn't match(ktsj)
  - I'd like to fix this specification before 2.7 release.

#13 - 11/24/2019 10:46 PM - Eregon (Benoit Daloze)
- [Misc #16188] The performance overhead of ruby2_keywords (erogen)
  - It's about 7-8% (mame/Yusuke Endoh measurements) for foo(*args) calls where foo has little code, both for MRI and TruffleRuby (I measured about 5-11% overhead). That's quite significant.
  - I would like to see a plan to avoid that significant overhead in Ruby 3+. I propose to either remove ruby2_keywords in 3.0, or make ruby2_keywords explicit with send_keyword_hash (removes the overhead), or use another way (e.g., ...) to delegate in 2.7+.
  - Non-lexical delegation cases seem rare, it seems unfortunate to slow down every foo(*args) for very few delegation cases which could use an extra send_keyword_hash.
- [Feature #16378] Support leading arguments together with ... (erogen)
  - Otherwise ... cannot be used in many cases.
  - I think it's what most people expect.
  - ... is a nice way to do delegation for lexical cases.

#14 - 11/24/2019 11:14 PM - mame (Yusuke Endoh)
- [Feature #16345] Don't emit deprecation warnings by default. (mame)
  - The discussion seems to be agreeing with deduplicated warnings [Feature #16289]. We must decide.

#15 - 11/25/2019 06:00 AM - sawa (Tsuyoshi Sawada)
- [Feature #16166] Remove exceptional treatment of *foo when it is the sole block parameter (sawa)
  - Unintended arity. This must be fixed in an earlier stage before Ruby 3.
- [Feature #16274] Transform hash keys by a hash (sawa)
  - Easily rename hash keys that do not follow a rule

#16 - 11/27/2019 01:06 AM - methodmissing (Lourens Naudé)
- [Misc #16375] Right size regular expression compile buffers for literal regexes and on Regexp#freeze (lourens)
  - Builds on Misc #16291, I think there's potential to apply this pattern to other types at hooks outlined at the end of the issue
  - A large set of literal regular expressions are quite common in Rails applications (mostly framework, but also application and dependencies)
  - In my Redmine boot test was able to shave 300kb off just excess regex buffer capacity

#17 - 11/27/2019 11:58 AM - hsbt (Hiromi SHIBATA)
- [Feature #15605] json library needs more frequent releases
  - Should we bump the json version to 2.2.1?

#18 - 11/27/2019 12:49 PM - zverok (Victor Shepelev)
Ugh, forgot one

- [Misc #16260] Symbol#to_proc behaves like lambda, but doesn't aknowledge it *(the title says it all)*

#19 - 11/28/2019 06:40 AM - byroot (Jean Boussier)

- [Feature #16377] Regexp literals should be frozen (byroot)
  - Previous discussion in [Feature #16345]
  - Regexp literals always return the same mutable object, it can cause state to leak.

#20 - 11/28/2019 10:15 AM - Eregon (Benoit Daloze)

- [Feature #16379] Backporting ... to Ruby 2.4 - 2.6 and pass_keywords (eregon)
  - I'd like to have the opinion of the core team on this.

#21 - 11/30/2019 04:06 AM - mame (Yusuke Endoh)
- Status changed from Open to Closed

The meeting has been already ended. I'll create a new ticket DevelopersMeeting20191220Japan later.