The next dev meeting

Date: 2021/11/18 13:00-17:00 (JST)
Place/Sign-up/Agenda/Log: https://github.com/ruby/dev-meeting-log/blob/master/DevelopersMeeting20211118Japan.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.

- It is recommended to add a comment by 2021/11/15. We hold a preparatory meeting to create an agenda a few days before the dev-meeting.
- The format is strict. We'll use this script to automatically create an markdown-style agenda. We may ignore a comment that does not follow the format.
- Your comment is mandatory. We cannot read all discussion of the ticket in a limited time. We appreciate it if you could write a short summary and update from a previous discussion.

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

---

History

#1 - 10/25/2021 09:11 AM - mame (Yusuke Endoh)
- Related to Misc #14770: [META] DevelopersMeeting added

#2 - 10/25/2021 09:17 AM - zverok (Victor Shepelev)

  - [Feature #18262] Enumerator::Lazy#partition (zverok)
    - Part of the effort to make Lazy more natural: #partition to return two lazy enumerators

#3 - 10/26/2021 02:28 AM - shugo (Shugo Maeda)

  - [Feature #12737] Module#defined_refinements (shugo)
    - It's good to have for debugging purposes. Users can see refinements defined in a module in irb etc. without reading code.
  - [Feature #14332] Module.used_refinements to list refinement modules (shugo)
    - It's good to have for debugging purposes. Users can see all activated refinements in a given scope.
#4 - 10/26/2021 10:39 AM - shugo (Shugo Maeda)

- [Bug #18270] Refinement#(extend_object,append_features,prepend_features) should be removed
  - Refinements are different from normal modules, so it's potentially dangerous to use refinements as mixins.
  - Why not deprecate them in Ruby 3.1, and remove them in Ruby 3.2.

#5 - 10/26/2021 02:44 PM - Dan0042 (Daniel DeLorme)

- [Feature #16252] Hash#partition should return hashes (dan0042)
  - More consistent with select/reject
  - More in line with existing usage (most people seem to convert the results to hashes)
  - More efficient than arrays of arrays
  - Mostly compatible

#6 - 11/04/2021 01:16 PM - Eregon (Benoit Daloze)

- [Misc #18285] NoMethodError#message uses a lot of CPU/is really expensive to call (ereg)
  - OK to do [issue](https://bugs.ruby-lang.org/issues/18285#note-11) and no longer call custom #inspect for NoMethodError?
  - Same for NameError?
- [Bug #18243] Ractor.make_shareable does not freeze the receiver of a Proc but allows accessing ivars of it (ereg)
  - How about this approach to fix it [issue](https://bugs.ruby-lang.org/issues/18243#note-9)?
  - I think passing a Proc with a non-shareable self to a Ractor is highly confusing semantically and in practice, and it's prone to bugs.

#7 - 11/05/2021 08:55 AM - byroot (Jean Boussier)

- [Feature #18273] Class#subclasses
  - Something we forgot to discuss as part of [Feature #14394].
  - There are cases where you only want direct subclasses, not the whole tree.
  - Can either be Class#subclasses like in Active Record.
  - Or could be a boolean parameter on descendants.

#8 - 11/05/2021 06:42 PM - Eregon (Benoit Daloze)

- [Feature #18287] Support nil value for sort in Dir.glob (ereg)
  - Let's be consistent for core methods and treat nil as false, or raise if not boolean.

#9 - 11/08/2021 06:01 PM - jeremyevans0 (Jeremy Evans)

- [Feature #6210] load should provide a way to specify the top-level module (jeremyevans0)
  - This is fairly easy to implement in a backwards compatible manner, and seems useful.
  - Are we OK adding this feature? If so, is the pull request acceptable?
- [Feature #11256] anonymous block forwarding (jeremyevans0)
  - This was accepted by matz (Yukihiro Matsumoto) four years ago, but never committed.
  - I've created a new working version based on patches from nobu (Nobuyoshi Nakada) and mame (Yusuke Endoh).
  - There should be no backwards compatibility issues, as the proposed syntax is currently invalid.
  - Is it OK to add this feature? If so, is the pull request acceptable?
- [Feature #11689] Add methods allow us to get visibility from Method and UnboundMethod object. (jeremyevans0)
  - As requested by matz (Yukihiro Matsumoto), I put together a patch for #public?, #private?, and #protected?.
  - Do we want to use this approach, or do we want to reconsider the #visibility method that returns a symbol?
  - If we want to use this approach, is the pull request acceptable?
- [Feature #12495] Make "private" return the arguments again, for chaining (jeremyevans0)
  - This seems useful, and the backwards compatible issues are quite small.
  - Are we OK adding this feature? If so, is the pull request acceptable?
- [Feature #16663] Add block or filtered forms of Kernel#caller to allow early bail-out (jeremyevans0)
  - Currently it is not possible to generate a partial backtrace without knowing how many frames you need up front.
  - This feature allows the production of partial backtraces without that knowledge.
  - For example, you can use it to return the only the first frame that meets some criteria.
  - There shouldn't be any backwards compatibility issues, as caller/caller_locations does not currently use a block.
  - Are we OK adding this feature? If so, is the pull request acceptable?

#10 - 11/10/2021 12:02 PM - ioquatix (Samuel Williams)

- [Bug #18296] Custom exception formatting should override Exception#full_message.
  - (mame: you need to write your comment)

#11 - 11/15/2021 10:31 PM - duerst (Martin Dürst)

- [Feature #18336] How to deal with Trojan Source vulnerability
For [A], Bidi characters, I think "wait and see" may be the right decision, but it should be explicit.

- [Feature #18337] Ruby allows zero-width characters in identifiers
  - Suggest to use https://www.unicode.org/reports/tr31/, to catch up with other languages.

#12 - 11/16/2021 01:24 AM - mame (Yusuke Endoh)
Warning: Please read the instruction carefully. You must briefly write what you want us to discuss as a comment. It is ideal if we can start discussing without reading the ticket itself.

#13 - 11/16/2021 08:55 AM - byroot (Jean Boussier)
[Feature #18339] GVL instrumentation API

- I'd like 3 new internal tracepoints: RUBY_INTERNAL_EVENT_GVL_ACQUIRE_ENTER, RUBY_INTERNAL_EVENT_GVL_ACQUIRE_EXIT, RUBY_INTERNAL_EVENT_GVL_RELEASE
- It would be also very useful if the number of waiting thread was part of the hook arguments.
- It would allow to instrument the GVL which is a key metric for threaded environments, and to tune concurrency in applications.
- Is the feature acceptable?
- Any performance concerns when no hook is registered?

#14 - 11/18/2021 02:32 PM - mame (Yusuke Endoh)
- Status changed from Open to Closed
- Description updated

#15 - 11/18/2021 02:35 PM - mame (Yusuke Endoh)
- Description updated