Ruby master - Misc #15996

DevelopersMeeting20190829Japan

07/11/2019 12:35 PM - mame (Yusuke Endoh)

Status:	Closed	
Priority:	Normal	
Assignee:		

Description

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: 2019/08/29 (Thu)
Time: 13:00-17:00 (JST)
Place and Sign-up: https://connpass.com/event/139231/
log: https://docs.google.com/document/d/1XypDO1crRV9uNg1_ajxkljVdN8Vdyl5hnz462bDQw34/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 (ko1)
  * 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

#1 - 07/11/2019 12:37 PM - mame (Yusuke Endoh)

I added the following note.

```
**A short summary of a ticket is strongly recommended. We cannot read all discussion of the ticket in a limite
d time.**
A proposal is often changed during the discussion, so it is very helpful to summarize the latest/current propo
sal, post it as a comment in the ticket, and write a link to the comment.
```

Please consider adding a short summary comment to the ticket you want us to discuss.

#2 - 07/11/2019 12:39 PM - mame (Yusuke Endoh)

Carry over.

- [Feature #11460] Unhelpful error message when naming a module with the same name as an existing class (nobu)
- [Misc #15805] Let memory sizes of the various IMEMO object types be reflected correctly (methodmissing)
  - Only the imemo_tmpbuf type's auxiliary malloc heap buffer is factored into obj_memsize_of, other imemo types also alloc on the heap
  - PR: https://github.com/ruby/ruby/pull/2140
- [Misc #15806] Explicitly initialise encodings on init to remove branches on encoding lookup (methodmissing)
  - Currently rb_enc_from_index and related methods explicitly check for encoding table init on each call - I think this should be initialised at boot time instead as even the simple case of loading the interpreter requires the encoding table to be loaded anyways (symbol.c)
  - PR: https://github.com/ruby/ruby/pull/2128

#3 - 07/11/2019 12:39 PM - mame (Yusuke Endoh)

- Related to Misc #14770: [META] DevelopersMeeting added

#4 - 07/23/2019 01:24 PM - nobu (Nobuyoshi Nakada)

- [Feature #12093] Eval InstructionSequence with binding

#5 - 07/23/2019 06:10 PM - jeremyevans0 (Jeremy Evans)

- [Bug #11326] Defining a writer as a Struct member allowed?
  - Should we ban struct members ending in =?
  - Do we also want to ban struct members ending in ! or ??

#6 - 07/29/2019 10:36 AM - Glass_saga (Masaki Matsushita)

- [Feature #15588] String#each_chunk and #chunks
  - name: #each_slice and #slices are better?
  - size as characters or bytes
  - should we support :strip parameter?
- [Feature #15553] Addrinfo.getaddrinfo supports timeout
  - I'd like to have some feedback for patch2.diff

#7 - 07/30/2019 04:32 AM - sikachu (Prem Sichanugrist)

- [Misc #15723] Reconsider numbered parameters
  - Reading from description, I think there are a few supporters of removing @1, @2 in favor of just having @ refer to the first argument.
  - I also believe having just @ is enough for major use cases, and for the readability of the code when multiple implicit receivers are used.
  - Proposal: Remove @1 and @2 and have only @ for the first argument in Ruby 2.7+

#8 - 07/30/2019 08:42 AM - ko1 (Koichi Sasada)

Triage:

- [Feature #15998] Allow String#-@ to deduplicate tainted string, but return an untainted one
  - taint removal schedule
- [Feature #15123] Enumerable#compact proposal
- [Feature #15854] Tracing instance variable assignment
- [Bug #9588] program name variables tainted
- [Feature #12802] Add BLAKE2 support to Digest
- [Feature #16001] Provide an alias to Kernel#caller_locations(1,1) and Kernel#caller(1,1)
- [Bug #7768] Inherited Array class missing
- [Bug #16007] Process.clock_getres matches the clock in practice for Process::CLOCK_{PROCESS,THREAD}_CPUTIME_ID FAILED fails on armv7hl
- [Feature #15975] Add Array#pluck

- [Bug #10463] :~@ and :!@ are not parsed correctly
- [Feature #10123] Named capture groups don't set local vars when matched using ~
- [Feature #16011] Digit grouping
- [Feature #16018] Add a way to deprecate methods
- [Bug #11935] Date.new returns inconsistent errors when passed invalid arguments
- [Bug #12230] Wrong exception raised when IO.read from a pipe is passed an offset on Windows
- [Bug #16020] Forbid if elsif without a condition
- [Bug #15244] Method #extname return empty string if filename is dot ('.')
- [Feature #15955] UnboundMethod#apply
- [Feature #15976] Add Array#overlap? for whether the intersection of 2 arrays is non empty?
- [Bug #15558] Should Exception#exception copy the backtrace?

if there is a time to check...

#9 - 07/30/2019 11:30 AM - nobu (Nobuyoshi Nakada)

- [Feature #15868] Implement File.absolute_path?

#10 - 07/31/2019 03:49 AM - nobu (Nobuyoshi Nakada)

- [Feature #16029] Expose fstring related APIs to C-extensions

#11 - 07/31/2019 09:45 AM - Eregon (Benoit Daloze)

- [Feature #15778] caller_locations(debug: true) to access bindings of the stack.
  - Can we introduce the API? If not, why not? If not, please propose a way to support such functionality on other Ruby implementations (e.g., JRuby, TruffleRuby).

#12 - 08/02/2019 12:08 AM - nobu (Nobuyoshi Nakada)

- [Feature #16035] Allow non-finalizable objects such as Integer, static Symbol etc in ObjectSpace::WeakMap
- [Feature #16038] Provide a public WeakMap which compare by equality rather than identity

#13 - 08/07/2019 03:04 AM - ko1 (Koichi Sasada)

- Description updated

Schedule was changed!!

Before: 2019/08/20 (Thu) 13:00-17:00 (JST)
After: 2019/08/29 (Thu) 13:00-17:00 (JST)

Thanks,
Koichi

#14 - 08/07/2019 03:06 AM - ko1 (Koichi Sasada)

- Subject changed from DevelopersMeeting20190820Japan to DevelopersMeeting20190829Japan

#15 - 08/07/2019 01:10 PM - nobu (Nobuyoshi Nakada)

This was a missed carry-over on the last month.

- [Bug #15908] Detecting BOM with non-UTF encoding

#16 - 08/11/2019 10:35 AM - zverok (Victor Shepelev)

- [Feature #14781] Enumerator.generate. Patch provided; Matz seem to have liked the idea and name proposed, can we have it in 2.7?
- [Feature #14784] Comparable#clamp with a range. Implementation details are discussed, but generally proposal seem to be acceptable? Can we have it in 2.7, should I provide a patch?

#17 - 08/13/2019 09:29 AM - mtsmfm (Fumiaki Matsushima)

- [Feature #15815] Add option to raise NoMethodError for OpenStruct (mtsmfm)
  - We can use Symbol#to_proc with enumerable stuff when we use OpenStruct instead of Hash. For example: OpenStruct.new(JSON.parse(users)).map(&:id)
  - But it can't prevent typo. For Hash, it has Hash#fetch to raise KeyError.

#18 - 08/14/2019 12:40 PM - nobu (Nobuyoshi Nakada)

- [Feature #16103] Make the dot-colon method reference frozen

#19 - 08/21/2019 02:06 PM - k0kubun (Takashi Kokubun)

- ~~[Misc #16112] Reduce the possibility of "expand tabs" commit occurrences~~
  - Never mind. Resolved before the MTG thanks to @jeremyevans's comment.

#20 - 08/22/2019 12:09 AM - mrkn (Kenta Murata)

- [Feature #15864] Proposal: Add methods to determine if it is an infinite range
  - I proposed some candidates from mathematical terms

#21 - 08/22/2019 12:47 AM - mame (Yusuke Endoh)

- [Feature#14183] "Real" keyword argument
  - Jeremy and I agreed with Jeremy's proposal. I'd like to ask for matz's final confirmation.

#22 - 08/23/2019 12:56 AM - mame (Yusuke Endoh)

- [Feature #16115] Keyword arguments from method calls or ignore extra hash keys in splat
  - It proposes a triple splat (foo(***opt)) to pass a keyword hash with filtering out unknown keywords. What do you think?

#23 - 08/23/2019 02:44 AM - mame (Yusuke Endoh)

- [Feature #15955] UnboundMethod#apply
  - The ticket proposes a shortcut to unbound_method.bind(obj).call(args...) without allocation of a Method object. There are some use cases, and looks reasonable to me. What do you think?

#24 - 08/23/2019 09:13 AM - osyo (manga osyo)

- [Feature #15991] Allow questionmarks in variable names
  - Proposal: Allow ? in variable names
  - In #5781, matz is explicitly against an instance variable that ends with ?
  - How about allowing only local variables ?

#25 - 08/23/2019 05:43 PM - zverok (Victor Shepelev)

- [Feature #16113] Partial application: fetch(urls).map(&JSON.:parse.w(symbolize_names: true))
- [Feature #16122] Struct::Value: simple immutable value object

#26 - 08/26/2019 10:11 PM - Dan0042 (Daniel DeLorme)

- [Feature #10829] Add to_proc method to the Array class
- [Feature #12115] Add Symbol#call to allow to_proc shorthand with arguments
- [Feature #15302] Proc#with and Proc#by, for partial function application and currying
- [Misc #15723] Reconsider numbered parameters
- [Feature #15897] it as a default block parameter
- [Feature #16113] Partial application
- [Feature #16120] Implicit block argument if block starts with dot-method call
  - All proposals above are different ideas for a block shorthand more versatile than (&:to_s)
  - Insatisfaction with numbered parameters solution
  - In #16120 I try to analyze the root issue common to all the above

#27 - 08/27/2019 10:23 AM - duerst (Martin Dürst)

Dan0042 (Daniel DeLorme) wrote:

> - [Feature #10829] Add to_proc method to the Array class
> - [Feature #12115] Add Symbol#call to allow to_proc shorthand with arguments
> - [Feature #15302] Proc#with and Proc#by, for partial function application and currying
> - [Misc #15723] Reconsider numbered parameters
> - [Feature #15897] it as a default block parameter
> - [Feature #16113] Partial application
> - [Feature #16120] Implicit block argument if block starts with dot-method call
>   - All proposals above are different ideas for a block shorthand more versatile than (&:to_s)

- Insatisfaction with numbered parameters solution
- In #16120 I try to analyze the root issue common to all the above

I know I shouldn't comment on individual issues here, but I haven't found a better way of commenting on this group of issues (maybe we should create a superissue): I think we should stop dealing with individual ideas of how to improve functional programming in Ruby and work out an overview of what's missing and how to address it, understanding that Ruby is first and foremost an OO language and we'll never get to the same point as Haskell or something similar.

#28 - 08/27/2019 11:04 AM - shevegen (Robert A. Heiler)

> I know I shouldn't comment on individual issues here

While I agree, one problem is that e. g. Dan0042 did not read the prior comments here - in
particular he linked in "[Misc #15723] Reconsider numbered parameters", which already was
linked in by sikachu. IMO, these issues become quite a lot of discussion work in general
and I am not sure matz has enough time to comment on every single one when people just
copy/paste new issues without looking whether others already linked these in. :\

Otherwise I agree with Martin's statement of "work out an overview of what's missing and
how to address it", in regards to the functional approach. I myself use ruby almost
exclusively from an OOP view (mostly); other ruby users seem to like a more functional
approach, such as zverok. Sometimes these styles seem to clash, in my opinion.

The dev-meeting discussions are great for others in the ruby community who are not e. g.
in japan, but this here is probably the longest individual issue tracker for any dev
meeting as of yet; I am not sure it is possible to discuss all of this in e. g. 4 hours.
But I digress too, apologies for that.

#29 - 08/27/2019 02:56 PM - Dan0042 (Daniel DeLorme)

My apologies if I did something inappropriate. I *did* realize there was some duplication, I just thought these issues should be considered part of a single "package". 100% agree with duerst (Martin Dürst), what he describes is what I was trying to do.

#30 - 08/28/2019 03:07 AM - mame (Yusuke Endoh)

Looks like we have too many topics at the next meeting. Please do not expect all tickets to be discussed. Thank you for your understanding.

#31 - 08/28/2019 08:17 AM - mame (Yusuke Endoh)

zverok (Victor Shepelev) wrote:

> - [Feature #14781] Enumerator.generate. Patch provided; Matz seem to have liked the idea and name proposed, can we have it in 2.7?
> - [Feature #14784] Comparable#clamp with a range. Implementation details are discussed, but generally proposal seem to be acceptable? Can we have it in 2.7, should I provide a patch?

The two tickets were discussed at the past meetings. Next time, could you give us more informative short summary instead of "can we have it in 2.7?" We need to spend time to read the ticket to grab the progress from the previous discussion.
For example, the first issue is mainly a naming issue, so it would be helpful to list the proposed names. In the second one, akr pointed out the spec issue. (Do not call it "implementation details.") "Raising an exception is proposed for clamp(1...2)" would be even helpful.

zverok (Victor Shepelev) Sorry for picking only your agenda, but for all people, please consider saving our time (especially matz's).

#32 - 08/28/2019 11:44 AM - zverok (Victor Shepelev)

mame (Yusuke Endoh) OK, sorry. I was not sure how the process looks internally (e.g. should I save the space and attention of the reader in agenda by just listing things shortly, or provide the full context in-place). Here are updated descriptions, just in case:

- [Feature #14781] Enumerator.generate: Matz seems to have liked the idea and name proposed; implementation patch provided; the name should be decided upon: either generate, or, maybe, produce (looking like more-or-less oppose to reduce)
- [Feature #14784] Comparable#clamp with a range. The issue generally agreed on, except for behavior with upper bound with excluding-end range. ArgumentError is proposed for this case.

#33 - 08/28/2019 01:27 PM - mame (Yusuke Endoh)

zverok (Victor Shepelev) Thank you!

#34 - 09/08/2019 12:41 PM - mame (Yusuke Endoh)

- Status changed from Open to Closed