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 (kol)
  * 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 2021/12/06. 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
- Related to Misc #14770: [META] DevelopersMeeting
I think this is a natural addition after the addition of anonymous block forwarding in #11256.

Is it OK to support this syntax? If so, is the patch acceptable?

Two possible method names are find_caller and each_caller.

- Each_caller optimizes for the case where the first matching frame will be returned.
- Each_caller is more general, always iterating unless there is a non-local exit (break/return).

Are either of these names acceptable? If so, which method name should be used?

Two possible method names are find_caller and each_caller.

- Each_caller optimizes for the case where the first matching frame will be returned.
- Each_caller is more general, always iterating unless there is a non-local exit (break/return).

Are either of these names acceptable? If so, which method name should be used?

Two possible method names are find_caller and each_caller.

- Each_caller optimizes for the case where the first matching frame will be returned.
- Each_caller is more general, always iterating unless there is a non-local exit (break/return).

Are either of these names acceptable? If so, which method name should be used?

Two possible method names are find_caller and each_caller.

- Each_caller optimizes for the case where the first matching frame will be returned.
- Each_caller is more general, always iterating unless there is a non-local exit (break/return).

Are either of these names acceptable? If so, which method name should be used?

Two possible method names are find_caller and each_caller.

- Each_caller optimizes for the case where the first matching frame will be returned.
- Each_caller is more general, always iterating unless there is a non-local exit (break/return).

Are either of these names acceptable? If so, which method name should be used?

Two possible method names are find_caller and each_caller.

- Each_caller optimizes for the case where the first matching frame will be returned.
- Each_caller is more general, always iterating unless there is a non-local exit (break/return).

Are either of these names acceptable? If so, which method name should be used?

Two possible method names are find_caller and each_caller.

- Each_caller optimizes for the case where the first matching frame will be returned.
- Each_caller is more general, always iterating unless there is a non-local exit (break/return).

Are either of these names acceptable? If so, which method name should be used?

Two possible method names are find_caller and each_caller.

- Each_caller optimizes for the case where the first matching frame will be returned.
- Each_caller is more general, always iterating unless there is a non-local exit (break/return).

Are either of these names acceptable? If so, which method name should be used?

Two possible method names are find_caller and each_caller.

- Each_caller optimizes for the case where the first matching frame will be returned.
- Each_caller is more general, always iterating unless there is a non-local exit (break/return).

Are either of these names acceptable? If so, which method name should be used?

Two possible method names are find_caller and each_caller.

- Each_caller optimizes for the case where the first matching frame will be returned.
- Each_caller is more general, always iterating unless there is a non-local exit (break/return).

Are either of these names acceptable? If so, which method name should be used?

Two possible method names are find_caller and each_caller.

- Each_caller optimizes for the case where the first matching frame will be returned.
- Each_caller is more general, always iterating unless there is a non-local exit (break/return).

Are either of these names acceptable? If so, which method name should be used?

Two possible method names are find_caller and each_caller.

- Each_caller optimizes for the case where the first matching frame will be returned.
- Each_caller is more general, always iterating unless there is a non-local exit (break/return).

Are either of these names acceptable? If so, which method name should be used?

Two possible method names are find_caller and each_caller.

- Each_caller optimizes for the case where the first matching frame will be returned.
- Each_caller is more general, always iterating unless there is a non-local exit (break/return).

Are either of these names acceptable? If so, which method name should be used?
● [Feature #18033] https://bugs.ruby-lang.org/issues/18033
  ○ Time.at("2021-12-09") instead of Time.new.
  ○ Other new methods: Time.try_convert or Time.iso.

● [Feature #18190] Split Random::Formatter from securerandom
  ○ Already ruby/securerandom#7 has been merged.

● [Feature #18183] make SecureRandom.choose public

#11 - 12/07/2021 06:43 AM - sawa (Tsuyoshi Sawada)
  ● [Feature #18331] Kernel.#Time (sawa)
    ○ Introduce Kernel.#Time [instead of/in addition to] extending Time.new as in #18033.
    ○ Similar methods like Kernel.#Integer are used to parse strings, and this is a good fit for Time.

#12 - 12/09/2021 12:30 PM - mame (Yusuke Endoh)
  - Status changed from Open to Closed
  - Description updated