Ruby - Feature #18982

Add an `exception: false` argument for Queue#push, Queue#pop, SizedQueue#push and SizedQueue#pop

08/29/2022 10:20 AM - byroot (Jean Boussier)

Status:	Closed	
Priority:	Normal	
Assignee:		
Target version:		

Description

This replaces [Feature #18965]

Currently these methods raise in three occasions:

- ThreadError(queue empty) for #pop in nonblock=true mode, and the operation would block.
- ThreadError(queue full) for SizedQueue#push in nonblock=true mode, and the operation would block.
- ClosedQueueError if trying to #push in a closed queue.

I see several reasons to prefer a nil return value.

- Queue is often used in conjunction with threads, so you have to be very careful not to rescue an unrelated ThreadError.
- Queue if often used for low level code, deep in the stack, so exceptions are costly.

I propose that passing exception: false would cause the method to return nil instead of raising in the three cases listed above.

The argument exception: true is consistent with various other methods such as IO#read_nonblock(exception: false).

Related issues:

Related to Ruby - Feature #18965: Further Thread::Queue improvements	**Rejected**

Associated revisions

Revision 60defe0a68a40d1b3225cf6b971ea195e19ae2e2 - 10/17/2022 02:56 PM - byroot (Jean Boussier)

thread_sync.c: Clarify and document the behavior of timeout == 0

[Feature #18982]

Instead of introducing an exception: false argument to have non_block
return nil rather than raise, we can clearly document that a timeout of 0
immediately returns.

The code is refactored a bit to avoid doing a time calculation in
such case.

Revision 60defe0a68a40d1b3225cf6b971ea195e19ae2e2 - 10/17/2022 02:56 PM - byroot (Jean Boussier)

thread_sync.c: Clarify and document the behavior of timeout == 0

[Feature #18982]

Instead of introducing an exception: false argument to have non_block
return nil rather than raise, we can clearly document that a timeout of 0
immediately returns.

The code is refactored a bit to avoid doing a time calculation in
such case.

Revision 60defe0a - 10/17/2022 02:56 PM - byroot (Jean Boussier)

thread_sync.c: Clarify and document the behavior of timeout == 0

[Feature #18982]

Instead of introducing an exception: false argument to have non_block
return nil rather than raise, we can clearly document that a timeout of 0
immediately returns.

The code is refactored a bit to avoid doing a time calculation in such case.

History

#1 - 08/29/2022 10:21 AM - byroot (Jean Boussier)

- Related to Feature #18965: Further Thread::Queue improvements added

#2 - 08/29/2022 10:23 AM - byroot (Jean Boussier)

Proposed patch: https://github.com/ruby/ruby/pull/6299

#3 - 08/29/2022 10:24 AM - Eregon (Benoit Daloze)

Sounds good :+1:

#4 - 08/30/2022 12:08 AM - shyouhei (Shyouhei Urabe)

+1 for avoiding exceptions but nil can be problematic? Because a closed queue would also return nil for pop. You cannot distinguish if a queue is closed or would just block.

#5 - 08/30/2022 12:15 AM - byroot (Jean Boussier)

> but nil can be problematic?

This was discussed in #18774:

- #pop on closed queue already returns nil with no exception. Queue.new.tap(&:close).pop # => nil
- nil make sense because you can use it in a loop such as while item = queue.pop.
- There's not really any other value that make sense.

Overall it's really not that complicated to handle the various cases in which nil is returned. I have an real world example here: https://github.com/Shopify/statsd-instrument/blob/0af8a1e2e74fc24d1de8088ee995940033c1fe42/lib/statsd/instrument/batched_udp_sink.rb#L111-L138.

#6 - 08/30/2022 12:42 AM - jeremyevans0 (Jeremy Evans)

shyouhei (Shyouhei Urabe) wrote in #note-4:

> +1 for avoiding exceptions but nil can be problematic? Because a closed queue would also return nil for pop. You cannot distinguish if a queue is closed or would just block.

We could add another keyword argument for the exception value, and have that value returned instead of raising an exception (the keyword argument would default to nil). I'm not sure if it's worth supporting that, but it is a simple approach.

#7 - 08/30/2022 08:36 AM - shyouhei (Shyouhei Urabe)

I don't think people want to exit from a while item = queue.pop loop because the queue would block. for (;;) { if (errno != EAGAIN) break; ... } is a C idiom (people often break from a loop on error, *except* EAGAIN).

#8 - 09/22/2022 04:50 AM - ko1 (Koichi Sasada)

Queue#pop(timeout:0) returns nil if Queue is empty. Is it enough? (and forget optional nonblock argument)

#9 - 09/22/2022 06:46 AM - byroot (Jean Boussier)

> Is it enough?

Interesting, I never thought of that.

I think I can work with that, but then I think this behavior should be documented and speced.

Because timeout: 0 in some APIs (not necessarily ruby) can mean no timeout.

#10 - 10/06/2022 06:47 AM - ko1 (Koichi Sasada)

> Because timeout: 0 in some APIs (not necessarily ruby) can mean no timeout.

Already the implementation do -> "Queue#pop(timeout:0) returns nil if Queue is empty" so no problem?

#11 - 10/06/2022 06:49 AM - byroot (Jean Boussier)

Works for me. I'll leave this ticket open until I add some more spec and documentation to the existing method.

#12 - 10/17/2022 02:56 PM - byroot (Jean Boussier)

- Status changed from Open to Closed

Applied in changeset git|60defe0a68a40d1b3225cf6b971ea195e19ae2e2.

---

thread_sync.c: Clarify and document the behavior of timeout == 0

[Feature #18982]

Instead of introducing an exception: false argument to have non_block
return nil rather than raise, we can clearly document that a timeout of 0
immediately returns.

The code is refactored a bit to avoid doing a time calculation in
such case.