Ruby master - Feature #17369

Introduce non-blocking `Process.wait`, `Kernel.system` and related methods.

12/05/2020 12:10 PM - ioquatix (Samuel Williams)

Status: Closed
Priority: Normal
Assignee: ioquatix (Samuel Williams)
Target version:

Description
https://github.com/ruby/ruby/pull/3853

This PR introduces optional hooks to the scheduler interface for handling Process.wait, Kernel.system and other related methods (waitpid, wait2, etc).

It funnels all methods through a new interface Process::Status.wait which is almost identical to Process.wait except for several key differences:

- The return value is a single instance of Process::Status.
- It does not set thread local $?.

This is necessary for keeping the scheduler interface simple (and side effects are generally bad anyway).

History

#1 - 12/05/2020 12:53 PM - Eregon (Benoit Daloze)

Does such code still work, with a scheduler?

```
'echo foo'
p $? # => #<Process::Status: pid 43525 exit 0>
```

If not, it seems a significant problem, as existing code would break with a scheduler.

Given the implementation in the test scheduler:

```ruby
def process_wait(pid, flags)
  # This is a very simple way to implement a non-blocking wait:
  Thread.new do
    Process::Status.wait(pid, flags)
  end.join
end
```

It sounds like you would need a way to set $? on the current Thread.

So that $? can be set for the caller.

I think that's fine to add.

I think $? should be Fiber-local, probably it's thread-local only for historic reasons.

Otherwise, just switching between Fibers (e.g., on IO) would expose the $? of other Fibers, which will lead to bugs.

I expect that change to cause extremely few compatibility issues ($~, etc are already fiber-local + frame-local).

#2 - 12/05/2020 12:57 PM - ioquatix (Samuel Williams)

Does such code still work, with a scheduler?

Yes.

It sounds like you would need a way to set $? on the current Thread.

Nope, it's handled by Process.wait and so on.

Otherwise, just switching between Fibers (e.g., on IO) would expose the $? of other Fibers, which will lead to bugs.

Agree, but we can't change this without potentially breaking existing code.
Also, is it okay that Process.last_status and Process.last_status= (hypothetical) are fiber local? Because Matz already said he was against class attributes that are actually fiber local (even if I agree in theory, excluding the fact that this is a breaking change).

I expect that change to cause extremely few compatibility issues ($-, etc are already fiber-local + frame-local).

Great, if Matz can approve the change, then we can implement it, but it’s separate from this PR, since this PR just makes the existing interface non-blocking.

#3 - 12/05/2020 01:22 PM - Eregon (Benoit Daloz)
I clarified with ioquatix (Samuel Williams), the code above should be end.value so it returns the Process::Status and system still sets it. Then the change sounds good to me.

#4 - 12/06/2020 06:40 AM - ioquatix (Samuel Williams)
- Description updated

#5 - 12/08/2020 08:19 PM - ioquatix (Samuel Williams)
Non-blocking Process.wait has been merged.

#6 - 12/09/2020 08:40 AM - naruse (Yui NARUSE)
Is this feature discussed with ko1 and nobu?
Also I suspect Matz’s approval is required for this change.

#7 - 12/10/2020 06:08 AM - matz (Yukihiro Matsumoto)
I am OK with Process::Status.wait. As far as I’ve heard the code quality needs upgrade.

Matz.

#8 - 12/10/2020 08:40 AM - naruse (Yui NARUSE)
- Target version set to 3.0

#9 - 12/25/2020 02:43 AM - naruse (Yui NARUSE)
- Target version deleted (3.0)

#10 - 12/26/2020 07:40 AM - ioquatix (Samuel Williams)
We introduced experimental feature and implemented non-blocking hook for Ruby 3.
More work is required here, but we didn’t make it in time for Ruby 3.0 - so we marked it as experimental.
We also need to implement rb_f_system in terms of rb_process_status_wait. Can someone else help with this?

#11 - 02/09/2021 09:07 PM - jeremyevans0 (Jeremy Evans)
- Backport deleted (2.5: UNKNOWN, 2.6: UNKNOWN, 2.7: UNKNOWN)
- Tracker changed from Bug to Feature

#12 - 06/22/2021 11:46 AM - ioquatix (Samuel Williams)
See https://github.com/ruby/ruby/pull/4595 which implements non-blocking Kernel#system.

#13 - 09/27/2021 04:01 AM - ioquatix (Samuel Williams)
The implementation is completed.
However, some parts are still pretty messy, including leaking $? process status.
We need to make $? fiber local.

```
irb(main):012:0> Fiber.new(system("false")).resume; pp $? #<Process::Status: pid 628235 exit 1>
=> #<Process::Status: pid 628235 exit 1>
irb(main):013:0> Fiber.new(system("true")).resume; pp $? #<Process::Status: pid 628241 exit 0>
=> #<Process::Status: pid 628241 exit 0>
```

However this might cause some issues in existing code.
Should we consider to deprecate $??

#14 - 10/11/2021 05:48 PM - Eregon (Benoit Daloze)
I think making $? Fiber-local makes sense, and unlikely to break anything. I don't see the need to deprecate $?, and it's certainly not worth the cost to migrate existing code to some other way to the get the status.

#15 - 10/21/2021 07:49 AM - nobu (Nobuyoshi Nakada)
Please file a new issue for fiber-local $?.

#16 - 10/21/2021 07:50 AM - nobu (Nobuyoshi Nakada)
- Status changed from Assigned to Closed