Ruby - Feature #17490

Rename RubyVM::MJIT to RubyVM::JIT

12/30/2020 05:45 AM - k0kubun (Takashi Kokubun)

Status:	Closed	
Priority:	Normal	
Assignee:		
Target version:		

Description

Background

These days it's sometimes said that CRuby may add another lightweight JIT. Leaving RubyVM::MJIT under such a situation might imply RubyVM::MJIT will not impact the future JIT, but I think MJIT.pause/MJIT.resume should impact all JITs by default if --jit is going to enable all the JITs. The current naming will be confusing, and I think these features should named consistently with --jit.

I also think, although this feature is for JIT developers anyway, we should not add many APIs to control JIT (for now I want JIT to be a feature where users don't need to think about tuning it, and having such APIs might end up letting people do that), and this naming change will contribute to discouraging APIs for a particular JIT.

Proposal

Have the same constant as RubyVM::JIT, deprecate RubyVM::MJIT from Ruby 3.1, and remove the old one in Ruby 3.2.

Impact

This impacts only [Feature #14830] RubyVM::MJIT.pause / RubyVM::MJIT.resume, which is basically for k0kubun's own testing.

Related issues:

Related to Ruby - Feature #18349: Let --jit enable YJIT on supported platforms	**Closed**

Associated revisions

Revision e1fee7f949cb6719122672fa1081c60984a5339f - 01/14/2021 06:46 AM - k0kubun (Takashi Kokubun)

Rename RubyVM::MJIT to RubyVM::JIT

because the name "MJIT" is an internal code name, it's inconsistent with
--jit while they are related to each other, and I want to discourage future
JIT implementation-specific (e.g. MJIT-specific) APIs by this rename.

[Feature #17490]

Revision e1fee7f949cb6719122672fa1081c60984a5339f - 01/14/2021 06:46 AM - k0kubun (Takashi Kokubun)

Rename RubyVM::MJIT to RubyVM::JIT

because the name "MJIT" is an internal code name, it's inconsistent with
--jit while they are related to each other, and I want to discourage future
JIT implementation-specific (e.g. MJIT-specific) APIs by this rename.

[Feature #17490]

Revision e1fee7f9 - 01/14/2021 06:46 AM - k0kubun (Takashi Kokubun)

Rename RubyVM::MJIT to RubyVM::JIT

because the name "MJIT" is an internal code name, it's inconsistent with
--jit while they are related to each other, and I want to discourage future
JIT implementation-specific (e.g. MJIT-specific) APIs by this rename.

[Feature #17490]

Revision 02ba0bda7e548fcc7245d246324e253e5e2fc96a - 12/16/2021 04:02 AM - k0kubun (Takashi Kokubun)

Remove RubyVM::JIT (#5275)

[Feature #18349] reverts [Feature #17490]

Revision 02ba0bda7e548fcc7245d246324e253e5e2fc96a - 12/16/2021 04:02 AM - k0kubun (Takashi Kokubun)

Remove RubyVM::JIT (#5275)

[Feature #18349] reverts [Feature #17490]

Revision 02ba0bda - 12/16/2021 04:02 AM - k0kubun (Takashi Kokubun)

Remove RubyVM::JIT (#5275)

[Feature #18349] reverts [Feature #17490]

History

#1 - 12/30/2020 05:49 AM - k0kubun (Takashi Kokubun)

- Description updated

#2 - 12/30/2020 03:04 PM - Eregon (Benoit Daloze)

If we'd consider RubyVM is CRuby-specific, then it seems fine.
However, that's rather unclear, and then the question is what should other Ruby implementations should do RubyVM::MJIT and its methods?
(other Ruby implementations might be forced to define RubyVM for compatibility at some point, it wouldn't surprise me)

I think Ruby users should anyway not need to use RubyVM::MJIT methods, so I prefer the more precise MJIT name.
RubyVM is already awfully confusing. RubyVM::JIT sounds like a general/portable API (from the name) when it is in fact CRuby-specific.

IMHO, a benchmark harness should not need to use a JIT-specific API, so I think pause/resume are only useful for debugging.

RubyVM::MJIT.enabled? could become a portable API, but it would need to move somewhere else.
I'm also unsure if it's needed besides MRI tests, in part because all major Ruby implementations already show if there is a JIT in RUBY_DESCRIPTION.

#3 - 12/30/2020 10:59 PM - nobu (Nobuyoshi Nakada)

Eregon (Benoit Daloze) wrote in #note-2:

> If we'd consider RubyVM is CRuby-specific, then it seems fine.

Yes.

39a43d9cd09f

```
/*
 * Document-class: RubyVM
 *
 * The RubyVM module only exists on MRI. +RubyVM+ is not defined in
 * other Ruby implementations such as JRuby and TruffleRuby.
```

#4 - 12/31/2020 07:47 AM - k0kubun (Takashi Kokubun)

> If we'd consider RubyVM is CRuby-specific, then it seems fine.
> However, that's rather unclear, and then the question is what should other Ruby implementations should do RubyVM::MJIT and its methods?
>
> RubyVM::MJIT.enabled? could become a portable API, but it would need to move somewhere else.
> I'm also unsure if it's needed besides MRI tests

Didn't you clarify it by yourself at [Feature #15743]? The person who wrote the line that nobu quoted was you. You made it pretty clear that RubyVM::MJIT doesn't need to exist in other implementations.

> I think Ruby users should anyway not need to use RubyVM::MJIT methods, so I prefer the more precise MJIT name.

So, would you suggest always explaining what is "M"JIT in every release note and renaming --jit to --mjit to approach the naming inconsistency issue explained in this ticket? I'm fine with pasting a link for the former, but I'm reluctant to force users to remember the name for the flag.

> IMHO, a benchmark harness should not need to use a JIT-specific API, so I think pause/resume are only useful for debugging.

I agree. None of such benchmarks are supposed to be used for purposes other than testing MRI vs MRI+MJIT. I mean, can't I even put a script to test

it?

#5 - 12/31/2020 08:05 AM - hsbt (Hiroshi SHIBATA)

I'm +1 to rename it.

#6 - 01/01/2021 12:20 PM - Eregon (Benoit Daloze)

k0kubun (Takashi Kokubun) wrote in #note-4:

> Didn't you clarify it by yourself at [Feature #15743]? The person who wrote the line that nobu quoted was you. You made it pretty clear that RubyVM::MJIT doesn't need to exist in other implementations.

I tried to document it. I don't think it's good enough, many people don't read class documentation.
If e.g., they see RubyVM::AbstractSyntaxTree in a blog post, they might use it and miss the fact it's MRI-specific, because the name gives no clue about it.
And it gets more complicated, as RubyVM::AbstractSyntaxTree could exist on other Ruby implementations without much issues, while RubyVM::InstructionSequence probably cannot.

> So, would you suggest always explaining what is "M"JIT in every release note

Actually I think MJIT is a lot clearer than JIT in e.g. https://www.ruby-lang.org/en/news/2020/12/25/ruby-3-0-0-released/
MJIT is not the only "Ruby Just-in-Time compiler", so being precise seems helpful to me when mentioning it by name.

> and renaming --jit to --mjit to approach the naming inconsistency issue explained in this ticket? I'm fine with pasting a link for the former, but I'm reluctant to force users to remember the name for the flag.

Agreed the flag should be --jit as it is.

Since users shouldn't use RubyVM::MJIT methods anyway, I think it doesn't matter much what is the constant name for users.
My concern is from the name, RubyVM::JIT sounds like some official "blessed" API ("the JIT of the Ruby VM", nothing in that name hints it's MRI-specific), while RubyVM::MJIT has a hint of "implementation details"/shouldn't be used by casual users.
Really, the fault is the bad name of the RubyVM constant which completely misses the most important thing which is to hint it's MRI specific in the name, but I guess that's a lost battle (#15743).

#7 - 01/01/2021 12:48 PM - Eregon (Benoit Daloze)

FYI, I filed #17500 regarding RubyVM.
I think ExperimentalFeatures::JIT would be fine if we accept that issue, and e.g. other Rubies might implement ExperimentalFeatures::JIT.enabled? then, which makes sense to me.

#8 - 01/02/2021 03:06 AM - k0kubun (Takashi Kokubun)

Now some discussions seem to be mixed, so let's sort them out:

RubyVM::

> I tried to document it. I don't think it's good enough, many people don't read class documentation.
> If e.g., they see RubyVM::AbstractSyntaxTree in a blog post, they might use it and miss the fact it's MRI-specific, because the name gives no clue about it.
> And it gets more complicated, as RubyVM::AbstractSyntaxTree could exist on other Ruby implementations without much issues, while RubyVM::InstructionSequence probably cannot.
>
> My concern is from the name, RubyVM::JIT sounds like some official "blessed" API ("the JIT of the Ruby VM", nothing in that name hints it's MRI-specific), while RubyVM::MJIT has a hint of "implementation details"/shouldn't be used by casual users.
> Really, the fault is the bad name of the RubyVM constant which completely misses the most important thing which is to hint it's MRI specific in the name, but I guess that's a lost battle (#15743).
>
> FYI, I filed #17500 regarding RubyVM.
> I think ExperimentalFeatures::JIT would be fine if we accept that issue, and e.g. other Rubies might implement ExperimentalFeatures::JIT.enabled? then, which makes sense to me.

Yes, let's move to #17500. I thought you would not care about whether it's WhateverMRISpecificName::MJIT or WhateverMRISpecificName::JIT too.

--jit

> Agreed the flag should be --jit as it is.

:+1:

MJIT.pause / .resume

> Since users shouldn't use RubyVM::MJIT methods anyway, I think it doesn't matter much what is the constant name for users.

Yeah, when this API was introduced, it was said that "There is no objection (because RubyVM is just for internal features)" https://bugs.ruby-lang.org/issues/14830#note-3. I expected a similar kind of feedback to this ticket and meant to only confirm that here, although it seems to have triggered other discussions.

Release notes

> Actually I think MJIT is a lot clearer than JIT in e.g. https://www.ruby-lang.org/en/news/2020/12/25/ruby-3-0-0-released/
> MJIT is not the only "Ruby Just-in-Time compiler", so being precise seems helpful to me when mentioning it by name.

Are you saying, if we write "JIT" in an MRI release note, readers might get confused about whether it's MRI's JIT or TruffleRuby's Graal?

Once we have another tier of JIT in MRI, I'd agree it's useful to mention the name "MJIT" at least in a NEWS like:

```
# JIT
## Lightweight JIT
...

## MJIT
...
```

but a release note is something which is read by many non-Rubyist people who often don't know there has been a JIT since Ruby 2.6 in the first place. Many people thought Ruby 3 newly introduced a JIT, reading the latest release note. Using our original jargon would simply confuse such audiences. I'm personally *okay* with mentioning MJIT in NEWS.md though.

#9 - 01/02/2021 03:07 AM - k0kubun (Takashi Kokubun)

- Description updated

#10 - 01/02/2021 12:50 PM - Eregon (Benoit Daloze)

> ## Release notes
>
> > Actually I think MJIT is a lot clearer than JIT in e.g. https://www.ruby-lang.org/en/news/2020/12/25/ruby-3-0-0-released/
> > MJIT is not the only "Ruby Just-in-Time compiler", so being precise seems helpful to me when mentioning it by name.
>
> Are you saying, if we write "JIT" in an MRI release note, readers might get confused about whether it's MRI's JIT or TruffleRuby's Graal?

Not really, I'm saying the "JIT" is just unclear, "which JIT is it?".
The 3.0.0 release notes are hard to read for me because they mix MJIT and JIT.
If we use MJIT everywhere in that note, then I think it is very clear.
If we use JIT everywhere in that note, then I think it is very unclear.
I show examples below.

> Once we have another tier of JIT in MRI, I'd agree it's useful to mention the name "MJIT" at least in a NEWS like:

Yes, I was also thinking about that, if there is another JIT in MRI then RubyVM::JIT would become ambiguous.

> but a release note is something which is read by many non-Rubyist people who often don't know there has been a JIT since Ruby 2.6 in the first place.

That means they didn't read previous release notes then?
So probably they read the 3.0 notes because of the new major version.

I think a single sentence, or a link for MJIT would be best, e.g.,
"MJIT is a method JIT for CRuby available since Ruby 2.6."

> Many people thought Ruby 3 newly introduced a JIT, reading the latest release note.

I think that's partly caused by using the term JIT and not MJIT in some places.

"Many improvements were implemented in MJIT." is very clear.

"As of Ruby 3.0, JIT is supposed to give performance improvements in limited workloads" is unclear, does it mean a JIT appeared in 3.0? (no)
Same with JIT->MJIT is clear (and it connects to the previous sentence):
"As of Ruby 3.0, MJIT is supposed to give performance improvements in limited workloads"

#11 - 01/02/2021 03:51 PM - k0kubun (Takashi Kokubun)

I get your points. Nothing is technically wrong in what you're saying, and as a committer like you it's easier to interpret MJIT.

But I mean, I assume we're not writing a release note for people like us who can just read details in NEWS but writing it for people who are just not interested in such details. Saying "JIT became faster" is enough to just get attention from various andiences (and explaining details which people are not interested in only hurts it), and if people really care about "which JIT is it?", they should go to NEWS and we should clarify it once we have multiple JITs.

#12 - 01/03/2021 01:28 PM - Eregon (Benoit Daloze)

JIT just means "Just-in-Time", so actually to be grammatically correct I guess the release note should either refer to "MJIT" (a proper noun) or "the JIT compiler (bundled with Ruby)".
Either way, I think to make it clear in the release note, the only way would be to add a sentence like
"Ruby ships with a JIT since Ruby 2.6 (named MJIT). In Ruby 3.0, it has been improved significantly".

I don't see why an internal/experimental module (RubyVM::MJIT) need to have name understandable by people knowing nothing about MJIT.
I'd say users of RubyVM::MJIT should at least know the name of the JIT before using those APIs.

#13 - 01/04/2021 04:50 AM - k0kubun (Takashi Kokubun)

This thread is already too long for off topics, so let me focus on the name of RubyVM::MJIT in this ticket and leave the release notes discussion to elsewhere.

> I don't see why an internal/experimental module (RubyVM::MJIT) need to have name understandable by people knowing nothing about MJIT.
> I'd say users of RubyVM::MJIT should at least know the name of the JIT before using those APIs.

If the feature will continue to be MJIT-specific, your point would make sense. But as I'm trying to make --jit and RubyVM::MJIT consistent in terms of their name, I think their feature should also be consistent; RubyVM::MJIT.pause/resume should control not only MJIT but also other JITs if --jit is going to enable all future tiers of JIT (at least by default, even if we add an option to specify a JIT to it). I think the number of JIT-relaed APIs should be as few as possible, and this change will contribute to limiting per-JIT features. I don't want RubyVM::YetAnotherJIT in the future.

By the way, let me point out again that this feature is mainly for MJIT's development / testing and not for people who don't contribute to developing MJIT. Please don't take my development time away for discussing this if you only care about TruffleRuby's compatibility with modules under RubyVM, because clearly there's no need to do so.

#14 - 01/04/2021 05:11 AM - k0kubun (Takashi Kokubun)

- Description updated

#15 - 01/13/2021 06:29 AM - matz (Yukihiro Matsumoto)

As of #15743, it was made clear that RubyVM means CRuby specific module (OK? @Eregon (Benoit Daloze)). I am OK with renaming it JIT (or not). So it's totally up to @k0kubun (Takashi Kokubun).

In addition, I don't think we need a migration path for this internal feature.

Matz.

#16 - 01/14/2021 06:50 AM - k0kubun (Takashi Kokubun)

- Status changed from Open to Closed

Applied in changeset git|e1fee7f949cb6719122672fa1081c60984a5339f.

Rename RubyVM::MJIT to RubyVM::JIT

because the name "MJIT" is an internal code name, it's inconsistent with
--jit while they are related to each other, and I want to discourage future
JIT implementation-specific (e.g. MJIT-specific) APIs by this rename.

[Feature #17490]

#17 - 01/17/2022 06:00 AM - znz (Kazuhiro NISHIYAMA)

MEMO: This feature reverted at https://github.com/ruby/ruby/commit/1a63468831524f68e73cbb068071652c6486cfc6 without major release.

#18 - 01/17/2022 06:32 AM - k0kubun (Takashi Kokubun)

The revert was proposed and implemented at https://bugs.ruby-lang.org/issues/18349.

#19 - 01/17/2022 06:32 AM - k0kubun (Takashi Kokubun)

- Related to Feature #18349: Let --jit enable YJIT on supported platforms added