Ruby master - Feature #13156

In-tree copy of ruby/spec

01/24/2017 09:05 PM - Eregon (Benoit Daloze)

Status: Closed
Priority: Normal
Assignee: matz (Yukihiro Matsumoto)
Target version:

Description
I would like to integrate an in-tree copy of ruby/spec under spec/rubyspec/
(at the same location ruby/spec is currently cloned to ease transition).
This way, implementation and tests can be changed in a single commit, just like with MRI tests.
I will keep synchronizing the changes to ruby/spec, so it can be used
directly and other people can contribute there.
I know this synchronization works because I have been doing it for over 2 years.
This process is also well documented:
https://github.com/ruby/spec/wiki/Merging-specs-from-JRuby-and-other-sources

My main motivation is the current status is suboptimal as
ruby/spec is a different repository and there are synchronization problems.
For example, if a change happens on MRI trunk and it needs to adapt
ruby/spec, it will pass either ruby/ruby CI or ruby/spec CI but never both.
I want to fix this and I want to encourage CRuby developers to write specs.

If you want more details, please look at my ruby-core post:
http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/78633

Please state you opinion, I would like to integrate this soon.

Associated revisions
Revision 95e8c48d - 05/07/2017 12:04 PM - Eregon (Benoit Daloze)
Add in-tree mspec and ruby/spec

- For easier modifications of ruby/spec by MRI developers.
- .gitignore: track changes under spec.
- spec/mspec, spec/rubyspec: add in-tree mspec and ruby/spec.
  These files can therefore be updated like any other file in MRI.
  Instructions are provided in spec/README.
  [Feature #13156] [ruby-core:79246]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@58595 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 58595 - 05/07/2017 12:04 PM - Eregon (Benoit Daloze)
Add in-tree mspec and ruby/spec

- For easier modifications of ruby/spec by MRI developers.
- .gitignore: track changes under spec.
- spec/mspec, spec/rubyspec: add in-tree mspec and ruby/spec.
  These files can therefore be updated like any other file in MRI.
  Instructions are provided in spec/README.
  [Feature #13156] [ruby-core:79246]

Revision 58595 - 05/07/2017 12:04 PM - Eregon (Benoit Daloze)
Add in-tree mspec and ruby/spec

- For easier modifications of ruby/spec by MRI developers.
- .gitignore: track changes under spec.
- spec/mspec, spec/rubyspec: add in-tree mspec and ruby/spec.
  These files can therefore be updated like any other file in MRI.
  Instructions are provided in spec/README.
  [Feature #13156] [ruby-core:79246]
Add in-tree mspec and ruby/spec

- For easier modifications of ruby/spec by MRI developers.
- .gitignore: track changes under spec.
- spec/mspec, spec/rubyspec: add in-tree mspec and ruby/spec. These files can therefore be updated like any other file in MRI. Instructions are provided in spec/README.

---

History

#1 - 01/24/2017 09:08 PM - Eregon (Benoit Daloze)
Was this discussed in the last developer meeting?

Sorry for making this feature request a bit late.

#2 - 01/24/2017 09:09 PM - Eregon (Benoit Daloze)
- Assignee set to Eregon (Benoit Daloze)

#3 - 01/25/2017 12:55 AM - duerst (Martin Dürst)
- Assignee changed from Eregon (Benoit Daloze) to matz (Yukihiro Matsumoto)

Benoit Daloze wrote:
Was this discussed in the last developer meeting?

Apparently not (see https://docs.google.com/document/d/1ZKk-vxoYko8b2H4mlZr4NhoHsi3GdZghNXNqBP2vTv9/pub). I plan to be at the next meeting, and I hope we can discuss it there.

I have changed the assignee to Matz, because first, he has to decide. If/when it's approved, we'll switch the assignee back to you.

#4 - 01/25/2017 01:23 AM - shyouhei (Shyouhei Urabe)
I'm neutral or somewhat positive to this. It sounds worth trying. At least I have no reason to object.

#5 - 01/25/2017 02:07 AM - ko1 (Koichi Sasada)
I'm not sure because I'm not a heavy github user, but most of github users feel happy on github contribution, doesn't it?

#6 - 01/25/2017 10:35 AM - Eregon (Benoit Daloze)
Koichi Sasada wrote:
I'm not sure because I'm not a heavy github user, but most of github users feel happy on github contribution, doesn't it?

Anyone can of course keep contributing on GitHub. But it's very inconvenient when changing something in a Ruby implementation to not be able to change tests/specs and implementation together. (For example, when behavior changes or more frequently when a new test/spec is added which failed before) Contributing on GitHub has also an overhead compared to just pushing to SVN for MRI committers.

We have the same approach in both JRuby and TruffleRuby, and I think it's much superior than having files in 2 separate repositories like with git submodules or the current approach in MRI. In other words, I want to make it as easy to add a test in ruby/spec than a test in test/ruby.

#7 - 02/22/2017 08:37 AM - akr (Akira Tanaka)
I recommend test-all over rubyspec because it is difficult to change developer's behavior. I.e. it is difficult to force CRuby committers to write new tests for new features in rubyspec.

If I remember correctly, jruby already uses test-all. So, it is possible to use test-all for ruby implementations other than CRuby. test-all may have room to improvement for other implementations, though.

#8 - 02/22/2017 09:34 AM - matz (Yukihiro Matsumoto)
- Status changed from Open to Feedback
I have several concerns:

- If ruby/spec tries to be a common executable specification of the language, it seems unnatural to have it in the repository of one particular implementation.

- Merging it would reduce synchronization cost. But I suspect your real intention is to make ruby/spec as a canonical test/spec suite, right? In that case, merging would not accomplish your real intention as some committers had shown their concerns about merging it, e.g. the difference between implementation test and executable spec, or difficulties of describing language behaviors using a relatively thick library (mspec) written in the language you are testing, etc.

In short, I am OK to merge it to the repository, as long as you are OK with above concerns.

Matz.

#9 - 02/22/2017 10:06 AM - Eregon (Benoit Daloze)

Yukihiro Matsumoto wrote:

I have several concerns:

- If ruby/spec tries to be a common executable specification of the language, it seems unnatural to have it in the repository of one particular implementation.

https://github.com/ruby/spec will remain as the canonical repository.

I want this in-tree copy for convenience and allowing to add specs for new features, regressions, etc without having to worry about complex synchronization issues.

- Merging it would reduce synchronization cost. But I suspect your real intention is to make ruby/spec as a canonical test/spec suite, right? In that case, merging would not accomplish your real intention as some committers had shown their concerns about merging it, e.g. the difference between implementation test and executable spec, or difficulties of describing language behaviors using a relatively thick library (mspec) written in the language you are testing, etc.

It would be great if more tests go into ruby/spec, but this is the choice of everyone working on MRI. I want to make it easier for those who are willing to add such specs.

In short, I am OK to merge it to the repository, as long as you are OK with above concerns.

Thank you, I will do it soon.

#10 - 02/22/2017 10:29 AM - Eregon (Benoit Daloze)

Akira Tanaka wrote:

I recommend test-all over rubyspec because it is difficult to change developer's behavior.

I.e. it is difficult to force CRuby committers to write new tests for new features in rubyspec.

I do not want to force anybody, just encourage writing new tests under spec/rubyspec for developers willing to try it.

If I remember correctly, jruby already uses test-all.

So, it is possible to use test-all for ruby implementations other than CRuby. test-all may have room to improvement for other implementations, though.

Yes, it is possible, and test-all is also used to some degree in TruffleRuby, but I can tell you from experience it is painful to work with it. (excluding is manual and excludes many assertions at a time for a single one failing, figuring what is the intended behavior of the test is really hard and time consuming, etc)

Some of these issues can be improved, but I think overall, without an explicit support for "documentaton of intended behavior", it will remain obscure what behavior some assertions are testing.

It also seems to me the test/unit style encourages testing many things at once, which is quite in opposition with the specs style and progressively implementing a feature.

Nevertheless, test-all is very useful for improving coverage and we should aim to make it easier for other implementations to reuse it. But if we can write some new tests in ruby/spec, I believe it will always be superior for other implementations and for documentation.

#11 - 02/24/2017 04:11 AM - duerst (Martin Dürst)
On 2017/02/22 19:29, eregontp@gmail.com wrote:

Issue #13156 has been updated by Benoit Daloze.

Nevertheless, test-all is very useful for improving coverage and we should aim to make it easier for other implementations to reuse it.

If you have any concrete ideas of how to do that, please tell us.

#12 - 05/07/2017 12:05 PM - Eregon (Benoit Daloze)
- Status changed from Feedback to Closed

Applied in changeset trunk|r58595.

Add in-tree mspec and ruby/spec

- For easier modifications of ruby/spec by MRI developers.
- .gitignore: track changes under spec.
- spec/mspec, spec/rubyspec: add in-tree mspec and ruby/spec. These files can therefore be updated like any other file in MRI.
  Instructions are provided in spec/README.
[Feature #13156] [ruby-core:79246]

#13 - 05/07/2017 12:36 PM - Eregon (Benoit Daloze)

I finally got around adding the in-tree copy of ruby/spec, sorry for the delay.

Dear MRI committers, you can now directly edit specs under spec/rubyspec like any other file in the repository. Your changes there and new specs are very welcome. This is even more important for new features, as it can help other Ruby implementations to implement them correctly and faster. It also documents the behavior.
A simple README is available at spec/README.md (https://github.com/ruby/ruby/tree/trunk/spec).

Any issue for contributing or missing documentation, etc is welcome to be reported on the issue tracker at https://github.com/ruby/spec/ or on this MRI bug tracker if you prefer.