Ruby - Feature #7549

A Ruby Design Process

12/12/2012 04:45 PM - brixen (Brian Shirai)

Status:	Rejected	
Priority:	Normal	
Assignee:		
Target version:		

Description

Matz,

At RubyConf 2012, I gave a talk about a design process for Ruby (http://www.confreaks.com/videos/1278-rubyconf2012-toward-a-design-for-ruby). So far, over 12,000 people have viewed that talk. I think it is reasonable to say that many people are concerned about and interested in a design process for Ruby.

On Monday, we had an IRC meeting of Ruby implementers. Most of the points in my proposal were discussed but I'm concerned that a lot of confusion remains.

I have written a post that describes a Ruby design process and hopefully clarifies points that people found confusing:

http://brixen.io/2012/12/11/a-ruby-design-process

I would like to propose this process for making changes to Ruby. I am going to put a summary of the process at http://rubyspec.org/design and ask for people who support the process to submit their signature. I'd like to request that you consider the response from the community for my proposal.

Thank you,
Brian

Related issues:

Related to Ruby - Feature #8271: Proposal for moving to a more visible, forma...	**Assigned**

History

#1 - 12/12/2012 05:14 PM - headius (Charles Nutter)

Correct link for the video: http://www.confreaks.com/videos/1278-rubyconf2012-toward-a-design-for-ruby

I will add my comments here a bit later. No major objections.

#2 - 12/12/2012 05:32 PM - naruse (Yui NARUSE)

> It does not need huge changes.

No, it needs change.
Continuous changes is Ruby.

> Wherever this proposed process seems heavy, it has been designed to produce good decisions and limit Ruby changes.

I absolutely object this.
Don't prevent changing Ruby.

> proposed process

I think what such solid process generate is not Ruby.

Additionally I can't understand why you can belive MRI will implement suggested features on the process,
and you don't imagine MRI implement new features as implementation specific.

= Theses

== thesis 1: Ruby is matz'

matz has absolute right over materials named Ruby.
Even if matz transfer some right of designing and implementing privilege to other people, matz can always override it.

What people can do is to persuade matz or fork Ruby.

== thesis 2: Ruby is changing

Ruby must aim for ideal Ruby.
Even if it breaks compatibility, Ruby should change itself with some transition period.

== thesis 3: Standard language of discussion

Standard language of discussion of designing Ruby is C and Ruby.
People can use English, Japanese and other natural languages for assistance.

#3 - 12/12/2012 08:23 PM - duerst (Martin Dürst)

It has been my experience in all kinds of situations (e.g. W3C,
IETF,...), and over well more than a decade, that:

- Adding process to something creates a lot of overhead,
  for little gain
- Getting process right is extremely difficult (way more difficult
  than getting technology right)
- Once some (however broken) process is introduced, it's very
  difficult to get rid of it again
- Design by committee in almost all cases leads to way suboptimal
  technology
  For people who haven't made this experience, it's usually difficult to
  believe. They have to make their own mistakes to understand. That's
  fine. But don't use Ruby to make these mistakes. Ruby deserves better.

In essence, with the current bug/future tracker, we are close to what
Python has with their PEPs, although we have way less overhead, which is
a good thing.

Regarding things such as parallelism, I know that Matz cares about it a
lot. If somebody has the talent, time, and energy to move away from the
GIL, that would be really great. Introducing a "design process",
however, will never solve that problem.

Getting signatures on a proposal with nice-sounding words doesn't solve
any problem either. It's not signs that count, it's code.

A programming language is a creative effort. It's not something like
access to education, where the word "equitable" applies. Nobody would
try to claim that they needed "equitable" access to Picasso's or
Mozart's masterworks.

Regards,   Martin.

On 2012/12/12 16:45, brixen (Brian Ford) wrote:

> Issue #7549 has been reported by brixen (Brian Ford).
>
> ---
>
> Feature #7549: A Ruby Design Process
> https://bugs.ruby-lang.org/issues/7549
>
> Author: brixen (Brian Ford)
> Status: Open
> Priority: Normal
> Assignee:
> Category:
> Target version:
>
> Matz,
>
> At RubyConf 2012, I gave a talk about a design process for Ruby (http://www.confreaks.com/videos/1278-rubyconf2012-toward-a-design-for-ruby). So far, over 12,000 people have viewed that talk. I think it is reasonable to say that many people are concerned about and interested in a design process for Ruby.
>
> On Monday, we had an IRC meeting of Ruby implementers. Most of the points in my proposal were discussed but I'm concerned that a lot of confusion remains.

> I have written a post that describes a Ruby design process and hopefully clarifies points that people found confusing:
>
> http://brixen.io/2012/12/11/a-ruby-design-process
>
> I would like to propose this process for making changes to Ruby. I am going to put a summary of the process at http://rubyspec.org/design and ask for people who support the process to submit their signature. I'd like to request that you consider the response from the community for my proposal.
>
> Thank you,
> Brian

#4 - 12/13/2012 12:23 AM - Anonymous

Hi,

I am against rigid rules that could easily bring us bureaucracy. I am
also against introducing "class" to the community member. All
community members (but me) have equal right. Some might be more
trustworthy from their past works, but as of right, they should be
equal.

Thus, instead of well-written rules, I propose:

- we will declare official place to submit feature proposal. It must
  be a kind of issue tracker to manage proposals. I prefer Ruby
  redmine we use now, but I am open to move to better one if there is.

- every proposal should be described in languages we understand,
  English, Ruby, or C. Probably English is the best.

- community members can show their opinion, vote, or persuade other
  members in the discussion follows the proposal. The "dictator" will
  make the decision. Community members still can persuade him to
  change his mind, with rationale.

- when the feature is accepted and merged into the "official" Ruby,
  the behavior should gradually be described by RubySpec. We may need
  more members to maintain RubySpec. And we may need to communicate
  more to distinguish "the spec" and the implementation dependent.

- appendix: community members can deprive of the power of the
  dictator, when he apparently loose his ability to make rational
  decision. After deprivation, the community will make up new rules,
  hopefully flexible ones. I don't want that situation, but the day
  will come sooner of later.

  ```
        matz.
  ```

#5 - 12/13/2012 12:49 AM - subwindow (Erik Peterson)

> What people can do is to persuade matz or fork Ruby.

An attitude like this is almost guaranteed to end in a fork. I do not want Ruby forked.

> Standard language of discussion of designing Ruby is C and Ruby.
> People can use English, Japanese and other natural languages for assistance.

How are we supposed to persuade matz and the people who have the most influence on him if the language of discussion is only spoken by 1.7% of the world's population? We have to utilize a lingua franca for us to communicate our ideas with each other. In this day and age that language must be English.

Personally I don't think of this proposal as a "heavy process." I see it as a starting ground for discussion on how to make the evolution of Ruby a more collaborative process. With the challenges that face Ruby in the next couple of years, collaboration is *essential* for the continued prosperity of Ruby proper.

#6 - 12/13/2012 01:23 AM - kosaki (Motohiro KOSAKI)

> > What people can do is to persuade matz or fork Ruby.
>
> An attitude like this is almost guaranteed to end in a fork. I do not want Ruby forked.
>
> > Standard language of discussion of designing Ruby is C and Ruby.
> > People can use English, Japanese and other natural languages for assistance.
>
> How are we supposed to persuade matz and the people who have the most influence on him if the language of discussion is only spoken by 1.7% of the world's population? We have to utilize a lingua franca for us to communicate our ideas with each other. In this day and age that language must be English.

Proper language depend on the communication place. example, I and
naruse-san talk by English when discussing on ruby-core. but I usually
use Japanese when I get lunch with matz or other japanese developers.

We are already discuss by English, right? You can't inhibit face to
face meeting language because it's totally meaningless. you anyway
don't listen such talk. That is language independent issue. I think.

If you want to inhibit face to face discussion itself, it doesn't make
sense too because each rubist naturally talk about ruby when they meet
and get a lunch. Nobody can stop it.

And your worry is just illusion. As far as I know, almost all feature
discussion was discussed on ruby-core. So no one can solve an
illusion.

> Personally I don't think of this proposal as a "heavy process." I see it as a starting ground for discussion on how to make the evolution of Ruby a more collaborative process. With the challenges that face Ruby in the next couple of years, collaboration is *essential* for the continued prosperity of Ruby proper.

#7 - 12/13/2012 01:39 AM - enebo (Thomas Enebo)

I consider this process to be way over the top. I can appreciate most of Brian's observations about problems in our process, but I really don't like the proposed solutions.

At the very least I feel like incrementally chipping away at our primary communication issues will have a decent payoff. For any process created, I primarily would like:

1. Reasonably detailed proposal with at least one implementation (implementation may lag original proposal to get feedback before starting)
   a. corner cases must be enumerated to best of ability
   b. consideration of how it can affect different implementations should be discussed in proposal

points a and b are not required to be perfect because a submitter obviously may not think of all corner cases or may not be able to actually know how all Rubies are implemented. This is ok provided that the proposal is created well in advance of its approval and deployment as a new feature of Ruby. "created well in advance" is vague because not all features require the same level of scrutiny or time to digest.

1. Some amount of regular implementer meetings to make sure all implementations are well aware of new proposals. This is also a place to talk about the many outstanding gaps between the Ruby implementations. As a part of process an Agenda item should eternally exist to cover new and existing proposals, so that issues can be exchanged.

At this point I would not go much further until we refine what a proposal for #1 is and use it/improve it (starting it Aarons example of wiki entry might be a good starting place). And see how much more process miscommunication improves by having #2 in place.

Language communication is another issue which is difficult. I think it is totally unreasonable for non-English speaking Japanese to be asked to learn to speak English if they want to continue being involved in Ruby. It is a laughable and a totally unrealistic suggestion (learning a new language takes years of serious effort). People suggesting this should really consider how insulting this is to people who have been working on Ruby for 10+ years. They made this language useful for us in the first place! Let's please try and find better middle ground (like getting some corporate sponsorship to help do 2-way translation or even a volunteer effort perhaps). Our implementers meetings have been in English with some translation by ebi and asarih, but we can hopefully get more communication from the Japanese-only developers at these meetings. I can see why this is a difficult problem, but it certainly is not an impossible problem.

#8 - 12/13/2012 01:53 AM - kosaki (Motohiro KOSAKI)

> Proper language depend on the communication place. example, I and
> naruse-san talk by English when discussing on ruby-core. but I usually

> use Japanese when I get lunch with matz or other japanese developers.
>
> We are already discuss by English, right? You can't inhibit face to
> face meeting language because it's totally meaningless. you anyway
> don't listen such talk. That is language independent issue. I think.
>
> If you want to inhibit face to face discussion itself, it doesn't make
> sense too because each rubist naturally talk about ruby when they meet
> and get a lunch. Nobody can stop it.
>
> And your worry is just illusion. As far as I know, almost all feature
> discussion was discussed on ruby-core. So no one can solve an
> illusion.

In short, *now* an official language of ruby-core is English and *now*
we are discussing by English. period.

#9 - 12/13/2012 02:00 AM - subwindow (Erik Peterson)

kosaki (Motohiro KOSAKI) wrote:

> We are already discuss by English, right? You can't inhibit face to
> face meeting language because it's totally meaningless. you anyway
> don't listen such talk. That is language independent issue. I think.
>
> If you want to inhibit face to face discussion itself, it doesn't make
> sense too because each rubist naturally talk about ruby when they meet
> and get a lunch. Nobody can stop it.

Sorry if I was not clear, but I absolutely do not want to inhibit face-to-face discussion. I'm speaking strictly of online discussion that forms the rationale and details of changes to Ruby.

> And your worry is just illusion. As far as I know, almost all feature
> discussion was discussed on ruby-core. So no one can solve an
> illusion.

I have no idea whether it is illusory or not. There is a lot of discussion in Japanese concerning changes to Ruby. I have no way of knowing how much of (for instance) ruby-dev makes its way onto ruby-core. I do not want to end Japanese-language discussion, but I believe a stronger effort should be made to translate those discussions into English so that the rest of the community can be informed.

I feel like this discussion of language is but a minor note in the larger (and more important) discussion of creating a more collaborative process for changes to Ruby.

#10 - 12/13/2012 02:38 AM - headius (Charles Nutter)

My turn!

I'll go point-by-point through the proposal.

1. Ruby Design Council

I like the idea of a central group through which feature discussions must pass. I do not support these discussions being closed or limited to members of that group.

My view is that this group is more to clarify the responsibilities of Ruby implementers rather than to ensure features are designed well. The latter would be a result of a good council, but the most important aspect here is getting new features or feature changes air time with important stakeholders.

Normal users are as important as council members, as matz says...but normal users and implementers have different responsibilities. A new feature may or may not affect a normal user, but ideally it should make them happier in the future. A new feature definitely affects all implementers, and ideally should not make them sadder. The council should be a mechanism to ensure that doesn't happen...not a mechanism to control the future of Ruby.

1. Proposals must be submitted by council members

Debatable, but not a bad idea. It would help guarantee council members are given the opportunity to weigh in (note I say opportunity here...any design committee needs to have a mechanism for abstaining). I'd say proposals can come from anywhere, but the council is gatekeeper (along with matz) for what would eventually be considered "standard".

1. Proposal criteria: English, complete, with RubySpecs

Not Japanese? Most of the folks implementing Ruby speak Japanese as their first language. If we're giving priority to implementers, the primary language should be Japanese.

Of course that's not tractable either. English is a good common language. I think the goal should be that all proposals are in English when they go to the council, but eventual docs should be in both English and Japanese. This might need to include RubySpec; have you (Brian) given any thought to localizing specs?

Requiring that an incoming proposal be complete is also intractable. Most of the issues of a proposal won't get fleshed out until discussed and implemented. It's not possible to cover every angle of a feature without discussion, and may not be possible without at least a reference impl. This has to be an organic, iterative process.

Requiring complete RubySpecs before implementation is also intractable. For a feature like refinements, how many hundreds of lines of specs would we need to write, never knowing if they're even valid until someone implements it? Again, this needs to be organic and iterative. RubySpecs or similar should be a required final artifact of the process, but it can't be a prerequisite for starting the process.

Brian does say in his longer description that he doesn't intend to stymie experimentation in the impls, and this process is not intended to stop impls from trying out feature ideas before proposing them.

1. Council members have veto power

I find the whole veto discussion offensive. We have been working on JRuby longer than any other alternative implementation, and never have I felt like matz didn't do the right thing when presented with the facts. I believe the council should have great weight in the decision to include or reject a feature, but we do not own Ruby. We are also biased; new features mean new work for us, so I have often fought the tendency to oppose any changes whatsoever. Our role is to help individual feature proposals evolve, not to control Ruby's evolution.

The council is to me more like a commission; a group of users responsible for accepting or making proposals, discussing and researching them thoroughly, and then making a recommendation upstream. That seems like the right model for Ruby.

1. All council members must implement approved features

Obviously we'll all want to be compatible with the "standard", but having this be a prerequisite to getting a feature into MRI ignores the fact that we all have resource constraints. Consider that no current implementation has fully implemented 1.9 features, and even JRuby -- the closest so far -- is still catching up. Should the process force us to drop everything and implement Ruby 2.1 features before we finish 1.9 or 2.0 work, just to allow those features to get into 2.1?

1. Discussion begins

This is completely unreasonable. At this point, you have required that the proposal be completely written up, with full specs (before any implementation starts and before the council sees the proposal), with implementations by all council members...all BEFORE discussion of the feature begins? This would require all implementations to implement half-baked features just to get them on the agenda. Do we really want a waterfall process for Ruby design?

Organic process...iterative...etc. Implementation (in a reference impl perhaps), discussion, and specification need to go hand-in-hand.

1. Final vote with veto privileges

Same comments as for #4.

--

Other concerns:

Requiring some new application to be written for this process to start is unreasonable. Can you provide an estimate for how long it will take to build such an app? Personally, I'd be happy with a new Redmine top-level project on MRI's install that we can follow (to go along with "ruby trunk", we'd have a "ruby design"). I am also not opposed to email and live discussions. I think the wiki-based "current view of the feature plus comments" is absolutely necessary. I do not see a need for a new app to be written, and I think waiting for such an app could derail the whole thing.

Honestly, since getting more involved in MRI's Redmine, it feels like it has everything we need. We just need to split off feature discussions and take better advantage of its project management features.

A separate mailing list would really help me, and for this process I think it's a must. Many to most discussions in ruby-core are specific to MRI (and I have been keeping up with all discussions for several weeks now).

I do not feel like RubySpec should be the sole language in which incoming proposals are tested, but I do feel like we should limit it. RubySpec would be preferred, but RSpec and test/unit should also be acceptable. It should be the responsibility of the council to see that RubySpecs get written during the process (perhaps by requiring it of the proposer).

I would like to see RubySpec governance expanded to the council members. No one member should control RubySpec if it is intended to become the one true standard Ruby compliance, or we're no better off than with MRI's test/*.

More to come as I ponder ramifications.

#11 - 12/13/2012 08:13 AM - shugo (Shugo Maeda)

matz wrote:

> I am against rigid rules that could easily bring us bureaucracy. I am
> also against introducing "class" to the community member. All
> community members (but me) have equal right. Some might be more
> trustworthy from their past works, but as of right, they should be
> equal.

Agreed.

I believe what we need is not a rigid process, but dialogues and discussions.

For example, Refinements will be limited and marked as experimental in 2.0, which is the result of discussions with Charles and some other guys. If you state a reason properly, you can change Ruby without such a rigid process.

It might be good to have a place dedicated to such discussions. It could be a new project of Redmine. However, only Matz should have veto power, because Ruby is his language.
My most serious worry is that Matz wouldn't be able to make his proposals of new features accepted by council members.
He said "code is documentation, where even bugs are described" before. In this context, "code" means an implementation, not test code. He often commits changes without tests:(

Speaking of documentation, I doubt it works well without an implementation.
I've written the spec of Refinements at https://bugs.ruby-lang.org/projects/ruby-trunk/wiki/RefinementsSpec, but there is no feedback about the documentation, except the one from Matz about typo.

I believe implementations, documentation, and tests should be developed complementarily with dialogues and discussions, and finally Matz should make decisions.

#12 - 12/13/2012 12:59 PM - naruse (Yui NARUSE)

2012/12/13 subwindow (Erik Peterson) erik@subwindow.com:

> > What people can do is to persuade matz or fork Ruby.
>
> An attitude like this is almost guaranteed to end in a fork. I do not want Ruby forked.

Imagine you have a great idea but matz reject it because "I feel the
design is not so good".
You believe the design is best because it is optimized for the use case,
but you cannot explain the use case in detail.

What you can is fork Ruby and distribute it for people who want the feature.
If people adopt it, matz will change the mind and merge the feature.

REE did it and some part of its feature are merged.

Of course if the feature can be distributed as gem, it need not fork.

> > Standard language of discussion of designing Ruby is C and Ruby.
> > People can use English, Japanese and other natural languages for assistance.
>
> How are we supposed to persuade matz and the people who have the most influence on him if the language of discussion is only spoken by 1.7% of the world's population? We have to utilize a lingua franca for us to communicate our ideas with each other. In this day and age that language must be English.

matz can read English, so you can use English.

--
NARUSE, Yui naruse@airemix.jp

#13 - 12/13/2012 01:29 PM - naruse (Yui NARUSE)

2012/12/13 subwindow (Erik Peterson) erik@subwindow.com:

> > And your worry is just illusion. As far as I know, almost all feature
> > discussion was discussed on ruby-core. So no one can solve an
> > illusion.
>
> I have no idea whether it is illusory or not. There is a lot of discussion in Japanese concerning changes to Ruby. I have no way of knowing how much of (for instance) ruby-dev makes its way onto ruby-core. I do not want to end Japanese-language discussion, but I believe a stronger effort should be made to translate those discussions into English so that the rest of the community can be informed.

Why don't you check ChangeLog/commit log and count new features only
discussed on ruby-dev?

--
NARUSE, Yui naruse@airemix.jp

#14 - 12/13/2012 01:53 PM - agarie (Carlos Agarie)

Hi,

> I do not want to end Japanese-language discussion, but I believe a
> stronger effort should be made to translate those discussions into English
> so that the rest of the community can be informed.

What about some mirror of ruby-dev automatically translated via google
translate? I've seen some sites with this feature. Probably the quality of
the translation won't be very good, but we might be able to "have an idea"
of what's being discussed there periodically.

> Why don't you check ChangeLog/commit log and count new features only
> discussed on ruby-dev?

This is a good idea, Naruse-san, but I think Erik wants to know what's
being discussed, not only the outcomes.

Carlos Agarie

Control engineering
Polytechnic School, University of São Paulo, Brazil
Computer engineering
Embry-Riddle Aeronautical University, USA

2012/12/12 NARUSE, Yui naruse@airemix.jp

> 2012/12/13 subwindow (Erik Peterson) erik@subwindow.com:
>
> > > And your worry is just illusion. As far as I know, almost all feature
> > > discussion was discussed on ruby-core. So no one can solve an
> > > illusion.
> >
> > I have no idea whether it is illusory or not. There is a lot of
> > discussion in Japanese concerning changes to Ruby. I have no way of knowing
> > how much of (for instance) ruby-dev makes its way onto ruby-core. I do not
> > want to end Japanese-language discussion, but I believe a stronger effort
> > should be made to translate those discussions into English so that the rest
> > of the community can be informed.
>
> Why don't you check ChangeLog/commit log and count new features only
> discussed on ruby-dev?
>
> --
> NARUSE, Yui naruse@airemix.jp

#15 - 12/13/2012 01:53 PM - naruse (Yui NARUSE)

2012/12/13 Carlos Agarie carlos.agarie@gmail.com:

> Hi,
>
> > I do not want to end Japanese-language discussion, but I believe a
> > stronger effort should be made to translate those discussions into English
> > so that the rest of the community can be informed.
>
> What about some mirror of ruby-dev automatically translated via google
> translate? I've seen some sites with this feature. Probably the quality of
> the translation won't be very good, but we might be able to "have an idea"

> of what's being discussed there periodically.
>
>> Why don't you check ChangeLog/commit log and count new features only
>> discussed on ruby-dev?
>
> This is a good idea, Naruse-san, but I think Erik wants to know what's being
> discussed, not only the outcomes.

My intention is that if you can check and count them, you can say
"topics of this kind should be in English" or something.
Or making some compromise like writing abstract in English on Japanese tickets.

--
NARUSE, Yui naruse@airemix.jp

#16 - 12/13/2012 01:59 PM - duerst (Martin Dürst)

Hello Erik,

On 2012/12/13 0:49, subwindow (Erik Peterson) wrote:

> Issue #7549 has been updated by subwindow (Erik Peterson).
>
>> What people can do is to persuade matz or fork Ruby.
>
> An attitude like this is almost guaranteed to end in a fork. I do not want Ruby forked.

So you are saying that you don't have the technical arguments to
persuade Matz? Maybe rather than having this process discussion, you
could just go and open a feature issue and describe the proposal you
have (if you have one).

>> Standard language of discussion of designing Ruby is C and Ruby.
>> People can use English, Japanese and other natural languages for assistance.
>
> How are we supposed to persuade matz and the people who have the most influence on him if the language of discussion is only spoken by 1.7% of the world's population? We have to utilize a lingua franca for us to communicate our ideas with each other. In this day and age that language must be English.

Sorry, I'm really wondering what you are talking about. Please correct
me if I'm wrong, but I haven't found any previous posts from you on
ruby-core. So I'm wondering why you complain about something not working
when you have rarely or never tried.

There are quite many people on ruby-core (English!) who have sent in
proposals via the tracker, and quite many of them got accepted. Of
course some others didn't get accepted, but there were good reasons for
that. Have you even tried to make a proposal? Or have you ever tried to
contribute to a proposal? Why do you complain that you have to use
Japanese when you were able to use English all the time but didn't give
it a try?

And while Japanese is only spoken by maybe 1.7% of the world's
population, it's spoken by a much higher percentage of Ruby committers.
But if you don't have anything to contribute on ruby-core (English), I'm
not sure why you think that ruby-dev (Japanese) is a problem. And if you
have something specific to ask about something on ruby-dev, why don't
you just go ahead and ask? It has happened in the past, and it usually
worked in that somebody made a summary of the issue or the discussion
was moved to English. But it's very inefficient to translate everything
on ruby-dev just for people who don't even make comments on ruby-core.

Having two lists is just a tradeoff. I read and write (and teach in)
Japanese, but I'm still much more efficient in English. Therefore I know
that for most of the Japanese participants on ruby-core, it's not
necessarily easy to contribute in English. Nevertheless, we see many and
good contributions in English. I have to say I really admire the
Japanese Ruby committers.

> Personally I don't think of this proposal as a "heavy process." I see it as a starting ground for discussion on how to make the evolution of Ruby a more collaborative process.

Why start with such a proposal before actually trying to collaborate?

> With the challenges that face Ruby in the next couple of years, collaboration is *essential* for the continued prosperity of Ruby proper.

Yes. But collaboration starts with collaboration, not with claiming that
collaboration is important before even having tried.

Regards, Martin.

> ---
>
> Feature #7549: A Ruby Design Process
> https://bugs.ruby-lang.org/issues/7549#change-34666
>
> Author: brixen (Brian Ford)
> Status: Open
> Priority: Normal
> Assignee:
> Category:
> Target version:
>
> Matz,
>
> At RubyConf 2012, I gave a talk about a design process for Ruby (
> http://www.confreaks.com/videos/1278-rubyconf2012-toward-a-design-for-ruby). So far, over 12,000 people have viewed that talk. I think it is reasonable to say that many people are concerned about and interested in a design process for Ruby.
>
> On Monday, we had an IRC meeting of Ruby implementers. Most of the points in my proposal were discussed but I'm concerned that a lot of confusion remains.
>
> I have written a post that describes a Ruby design process and hopefully clarifies points that people found confusing:
>
> http://brixen.io/2012/12/11/a-ruby-design-process
>
> I would like to propose this process for making changes to Ruby. I am going to put a summary of the process at http://rubyspec.org/design and ask for people who support the process to submit their signature. I'd like to request that you consider the response from the community for my proposal.
>
> Thank you,
> Brian

#17 - 12/13/2012 02:53 PM - duerst (Martin Dürst)

On 2012/12/13 13:35, Carlos Agarie wrote:

> Hi,
>
>> I do not want to end Japanese-language discussion, but I believe a
>> stronger effort should be made to translate those discussions into English
>> so that the rest of the community can be informed.
>
> What about some mirror of ruby-dev automatically translated via google
> translate?

Good idea. Why don't you set up such a mirror? It doesn't have to be on
the same site.

> I've seen some sites with this feature. Probably the quality of
> the translation won't be very good, but we might be able to "have an idea"
> of what's being discussed there periodically.

Yes indeed. You can even do this manually.

>> Why don't you check ChangeLog/commit log and count new features only
>> discussed on ruby-dev?

> This is a good idea, Naruse-san, but I think Erik wants to know what's
> being discussed, not only the outcomes.

There are thousands of different mailing lists and billions of Web pages in languages other than English. There are not enough people to translate them all. I get ruby-dev mails, and I can read them all if I want, but most of them are not really that important. It doesn't make sense to translate them all just because somebody might want to read some of them.

Regards,   Martin.

#18 - 12/14/2012 01:23 AM - trans (Thomas Sawyer)

Personally, I believe the No.1 factor for the better evolution or Ruby is for Ruby to be written in Ruby. Which basically means that Rubinius needs to become the primary implementation instead of MRI. Scoff if you will, but who would disagree that "eating your own dog food" is pretty vital to long term sustainability? I have also been a little concerned that development of Ruby is dominated by "C-think" b/c those who develop it are first and foremost C coders. To them Ruby is a secondary language (mostly used for systems admin scripting?).

If Ruby were written in Ruby, then actual Rubyists could participate in her development. They could actually implement some of the features they ask for rather then simply beg for them --and gems could handle much more in the way of extending the language. So if Matz rejects an idea, its not as bad b/c a 3rd party gem would be more likely to be able to offer the feature. I also think it would be somewhat eye-opening for current core developers. I think they'd gain a better perspective on Ruby itself, having to do almost everything in Ruby, which would bode even better for Ruby's future.

#19 - 12/14/2012 01:57 AM - jonforums (Jon Forums)

@trans I find your comment distracting and entirely off-topic. Brian opened with "I would like to propose this **process** for making changes to Ruby" and said nothing about specific implementations. Start a new thread on ruby-talk if you want to discuss the pros/cons of Rubinius, but please stay on-point if you feel you've got something of value to add. This topic is too important to litter with tangential discussions.

#20 - 12/14/2012 02:57 AM - trans (Thomas Sawyer)

@jonforums I disagree. It may not be in the same branch of conversation as preceding posts but I do not think it tangential. If the main line of Ruby development were in Ruby itself it would greatly effect this **process**, probably much more than anything else that's been discussed or agreed to thus far. Moreover, if history tells us anything, we've see this **process** discussion before (aka RCR). And like before, it won't amount to much.

#21 - 12/14/2012 06:19 AM - headius (Charles Nutter)

What language Ruby is implemented in is irrelevant to discussions about a feature design process going forward. It might help that process if the implementations of proposed features could be done entirely in Ruby (easier to read and comment on than C or Java, perhaps), but it doesn't change the process in any way.

Rubinius also wouldn't help for a lot of these features, since many of the interesting features would require changes to the C++ VM anyway (refinements, for example).

#22 - 12/14/2012 05:42 PM - naruse (Yui NARUSE)

As headius said, I think Rubinius is an implementation which is built with C++, not Ruby.

Anyway, this thread needs following:

- Definition of terms: some use the word "Ruby" for MRI, and others use for the new Ruby Standard. It is misleading.
- Just Cause: What is the aim of this? Who is the beneficiaries of this suggestion and what is the benefit?
- Product: what is the resulted product of this process, RubySpec? Documents? Mail log?

#23 - 12/17/2012 04:13 PM - nathany (Nathan Youngman)

@brixen I feel that your proposed process is requesting too many changes all at once. We need to take small steps.

I humbly suggest that we rely on prior works. Let's borrow from our neighbours, by adapting the processes other language communities use successfully. Then trim them down to what we can reasonably implement.

To that end, I suggest that we all read through the "Python Enhancement Proposal: Purpose and Guidelines". It could provide a good starting point, with written proposals that may be accepted or rejected, but without changing the role of Matz as BDFL. http://www.python.org/dev/peps/pep-0001/

A summary and a few more thoughts are posted to my blog:
http://nathany.com/ruby-design

Nathan
just some guy who enjoys Ruby :-)

#24 - 12/18/2012 03:44 AM - shyouhei (Shyouhei Urabe)

Right now, the easiest way to change ruby is to change ruby, without asking anything.

Almost all feature requests are silently ignored.   Others are explicitly rejected. I don't think a feature request sequence like matz says works. It isn't working already.

#25 - 12/18/2012 03:59 AM - trans (Thomas Sawyer)

=begin
@headius (Charles Nutter) @naruse (Yui NARUSE) Regardless of what might be thought of my idea, at least this much should be clear. Directly from the Rubinius home page:

((*"A large aspect of popular languages such as C and Java is that the majority of the functionality available to the programmer is written in the language itself. Rubinius has the goal of adding Ruby to that list. Rubyists could more easily add features to the language, fix bugs, and learn how the language works. Wherever possible Rubinius is written in Ruby. Where not possible (yet), it's C++."*))

And...

((*"The Rubinius bytecode virtual machine is written in C++, incorporating LLVM to compile bytecode to machine code at runtime. The bytecode compiler and vast majority of the core classes are written in pure Ruby."*))
=end

#26 - 12/21/2012 12:49 PM - naruse (Yui NARUSE)

nathany (Nathan Youngman) wrote:

> @brixen I feel that your proposed process is requesting too many changes all at once. We need to take small steps.
>
> I humbly suggest that we rely on prior works. Let's borrow from our neighbours, by adapting the processes other language communities use successfully. Then trim them down to what we can reasonably implement.
>
> To that end, I suggest that we all read through the "Python Enhancement Proposal: Purpose and Guidelines". It could provide a good starting point, with written proposals that may be accepted or rejected, but without chagoalnging the role of Matz as BDFL.
> http://www.python.org/dev/peps/pep-0001/

We had already tried to adopt some process like PEP, but it failed.
It is because of what headius said in [ruby-core:50838] "6. Discussion begins".

You may know, we CRuby people often discuss feature with patch.
With patch we can use a proposed feature by hand and design better API.
PEP is not suitable for such situation, so we finally adopt Redmine based process.
https://bugs.ruby-lang.org/projects/ruby/wiki/HowToRequestFeatures

This process is more open and everyone can propose feature.

> > A summary and a few more thoughts are posted to my blog:
> > http://nathany.com/ruby-design
>
> Wouldn't it be nice to see Matz's proposal for refinements, and its subsequent acceptance – or rejection. :-P

Did you read https://bugs.ruby-lang.org/issues/4085 ?

> We need a comprehensive language reference. Is the Ruby Core Reference the closest thing we have to the Python Language Reference? :-(

The Ruby Core Reference is generated from MRI's rdoc.

> But we gain the benefits of document driven design.

Newly added feature for MRI shall have rdoc even if some present features doesn't have.

zzak is adding documents for such APIs, and drbrain and other collaborators did Ruby 1.9.3 Documentation Challenge.
http://blog.segment7.net/2011/05/09/ruby-1-9-3-documentation-challenge
They are great work and you can start your work from the point.

#27 - 12/21/2012 04:56 PM - naruse (Yui NARUSE)

headius (Charles Nutter) wrote:

> 1. Ruby Design Council

> I like the idea of a central group through which feature discussions must pass. I do not support these discussions being closed or limited to members of that group.
>
> My view is that this group is more to clarify the responsibilities of Ruby implementers rather than to ensure features are designed well.

I felt this is straightforward one.
Ruby Design Process by brixedn didn't explicitly show its goal,
but I think its goal is to ease make a MRI compatible implementation.

> The latter would be a result of a good council,

As far as I understand, a council is not for good design but for consensus and compromise.
It sometimes causes bad design because of compromise.

> but the most important aspect here is getting new features or feature changes air time with important stakeholders.

MRI has Release Engeneering process, and you'd read mame's announce "2.0.0 feature freeze".
http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/48191
This is the air time with all stakeholders.

> Normal users are as important as council members, as matz says...but normal users and implementers have different responsibilities. A new feature may or may not affect a normal user, but ideally it should make them happier in the future. A new feature definitely affects all implementers, and ideally should not make them sadder. The council should be a mechanism to ensure that doesn't happen...not a mechanism to control the future of Ruby.

So what you want is a council for implementers, right?
And the goal for the council is to prevent adding a feature which is hard to implement, I guess.

> 1. Proposals must be submitted by council members
>
> Debatable, but not a bad idea. It would help guarantee council members are given the opportunity to weigh in (note I say opportunity here...any design committee needs to have a mechanism for abstaining). I'd say proposals can come from anywhere, but the council is gatekeeper (along with matz) for what would eventually be considered "standard".

If this council is for implementers, it sounds reasonable.

> 1. Proposal criteria: English, complete, with RubySpecs
>
> Not Japanese? Most of the folks implementing Ruby speak Japanese as their first language. If we're giving priority to implementers, the primary language should be Japanese.
>
> Of course that's not tractable either. English is a good common language. I think the goal should be that all proposals are in English when they go to the council, but eventual docs should be in both English and Japanese. This might need to include RubySpec; have you (Brian) given any thought to localizing specs?

If the goal of the council is written above, the language should be off course English.

> Requiring that an incoming proposal be complete is also intractable. Most of the issues of a proposal won't get fleshed out until discussed and implemented. It's not possible to cover every angle of a feature without discussion, and may not be possible without at least a reference impl. This has to be an organic, iterative process.

I think brixen's process is inspired by W3C HTML/CSS's process.
Seeing their process, a proposal should be comming with its implementation.
And after discussion and with some another implementations, it becomes a concreate spec.

> Requiring complete RubySpecs before implementation is also intractable. For a feature like refinements, how many hundreds of lines of specs would we need to write, never knowing if they're even valid until someone implements it? Again, this needs to be organic and iterative. RubySpecs or similar should be a required final artifact of the process, but it can't be a prerequisite for starting the process.
>
> Brian does say in his longer description that he doesn't intend to stymie experimentation in the impls, and this process is not intended to stop impls from trying out feature ideas before proposing them.

I think so.
brixen's process is not based on current/real development model.
It will prevent Ruby developing or simply ignored.

> 1. Council members have veto power

> I find the whole veto discussion offensive. We have been working on JRuby longer than any other alternative implementation, and never have I felt like matz didn't do the right thing when presented with the facts. I believe the council should have great weight in the decision to include or reject a feature, but we do not own Ruby. We are also biased; new features mean new work for us, so I have often fought the tendency to oppose any changes whatsoever. Our role is to help individual feature proposals evolve, not to control Ruby's evolution.

I'm glad that you think so.

> The council is to me more like a commission; a group of users responsible for accepting or making proposals, discussing and researching them thoroughly, and then making a recommendation upstream. That seems like the right model for Ruby.

It sounds like an Executive Commitee of the Java Community Process (JCP).
http://jcp.org/en/participation/committee

Its goal seems different from brixen's proposal.

> Other concerns:
>
> Requiring some new application to be written for this process to start is unreasonable. Can you provide an estimate for how long it will take to build such an app? Personally, I'd be happy with a new Redmine top-level project on MRI's install that we can follow (to go along with "ruby trunk", we'd have a "ruby design"). I am also not opposed to email and live discussions. I think the wiki-based "current view of the feature plus comments" is absolutely necessary. I do not see a need for a new app to be written, and I think waiting for such an app could derail the whole thing.

I setup a new project.
https://bugs.ruby-lang.org/projects/common-ruby

#28 - 12/21/2012 06:23 PM - duerst (Martin Dürst)

> Issue #7549 has been updated by naruse (Yui NARUSE).
>
> I setup a new project.
> https://bugs.ruby-lang.org/projects/common-ruby

Very good. But I think this could benefit from some followup/additional
text.

On the Redmine home page (https://bugs.ruby-lang.org/), please add:

How to propose a feature

Create an issue in project common-ruby.
<<<<

On the CommonRuby page, change:

A common set of Ruby(?). Don't report MRI dependent bugs here.

to:

A place to propose new features for Ruby (i.e. all Ruby
implementations). Don't report MRI dependent bugs here.

Some additional text may also help, e.g. something like:

Please describe why you want the new feature (uses cases preferred over
general reasoning), and how the feature should work
(implementations/specs/tests appreciated).

There are probably some more places that can benefit from some
adjustment (including Japanese versions), in particular also on
ruby-lang.org. Please tell me if you need some text.

Regards,   Martin.

#29 - 12/21/2012 11:23 PM - usa (Usaku NAKAMURA)

- Project changed from Ruby to 14

#30 - 01/01/2013 06:33 AM - brixen (Brian Shirai)

I've written another post addressing many of the misunderstandings about my proposal expressed in this thread:

http://brixen.io/2012/12/30/a-ruby-design-process-talking-points

We are starting work on the Consensus application here:

https://github.com/rubyspec/consensus

Cheers,
Brian

#31 - 01/04/2013 03:59 PM - duerst (Martin Dürst)

On 2013/01/01 6:33, brixen (Brian Ford) wrote:

> Issue #7549 has been updated by brixen (Brian Ford).
>
> I've written another post addressing many of the misunderstandings about my proposal expressed in this thread:
>
> http://brixen.io/2012/12/30/a-ruby-design-process-talking-points

This says:

"In fact, English is the language in which the Ruby ISO standard is
written. And that standard was written by MRI Japanese developers."

That standard was developed as JIS X 3017, in Japanese. Only later it
was translated to English for submission to ISO.

Regards,   Martin.

#32 - 01/04/2013 04:53 PM - duerst (Martin Dürst)

Hello Brian,

On 2013/01/01 6:33, brixen (Brian Ford) wrote:

> Issue #7549 has been updated by brixen (Brian Ford).
>
> I've written another post addressing many of the misunderstandings about my proposal expressed in this thread:
>
> http://brixen.io/2012/12/30/a-ruby-design-process-talking-points

This says:

The proposed design process attempts to reduce as much as possible the
need for all implementers to discuss proposed language features. The
discussion occurs after clear documentation is written, after precise
RubySpecs are written, and after everyone implements the feature so that
it passes RubySpec. The discussions then focus on concrete facts about
the impact of the proposed feature to existing and future code, whether
it is in libraries, frameworks, or applications.

This is the part of your proposal that I understand least (there are
other parts that I disagree quite strongly with, but at least I think I
understand why you proposed them).

What's the point of discussion if all implementations have already
implemented the feature? Discussion makes much more sense to me in the
early stages of some idea. Often somebody has an idea, but it's not

complete. There are many cases of feature proposals in redmine that
don't give all the details because the proposer hasn't figured out all
of them. But nevertheless, many of these proposals (of course not all)
have a lot of merit. One very clear category is where Matz has already
agreed with everything except that we are still missing a really good name.

Creating and evolving a programming language is a creative process. Such
a creative process is messy and takes time. Putting it in a process
straightjacket doesn't help.

This creative process is also where Matz is way ahead of everybody else.
He has been working on it for 20 years, and the results so far have been
to everybodys liking.

Giving every Ruby implementation the same weight of opinion may be
appropriate when it comes to implementability questions. Every serious
Ruby implementation has an established track record on implemenatability
issues. But when it comes to language design issues, nobody can in any
way match Matz's track record on language design.

At the end of your post, you write:

> The proposed design process seeks to create these three things:
>
> 1. A definition of the Ruby language that is independent of any particular implementation of Ruby.

That wouldn't be a bad thing to have.

> 1. A definition of Ruby that is explicit and verifiable by running RubySpec.

I think you also know that executable specs/tests are never able to
verify/prove that an implementation is correct/conformant. So this is
just impossible, unfortunately.

> 1. A process of deciding what features are in the Ruby language that is fair to all implementations of Ruby.

See above for this point. Fairness to implementations is appropriate for
implementability concerns (see e.g. the recent feedback from Charles
Nutter on implementation problems with refinements, which resulted in
quite some changes). Fairness to implementations isn't very relevant
when it comes to general language design issues.

Regards,   Martin.

#33 - 01/16/2013 12:29 PM - naruse (Yui NARUSE)

> http://brixen.io/2012/12/30/a-ruby-design-process-talking-points

It says "The purpose of the proposed design process is to prioritize those
things that create a unified Ruby programming language.".
But I think it is not the fundamental purpose.
Just after the sentence, "For businesses and developers whose salaries
derive from writing Ruby, the value of a unified Ruby programming language
should be obvious." must be your true desire, isn't it?

Your proposal is misunderstood because you don't write this and your
proposal is not based on this.
You should write the proposal based on the fundamental desire and its logic
and method must always be derived from that.
And it should have merits for people other than businesses and developers
whose salaries derive from writing Ruby.

2013/1/1 brixen (Brian Ford) brixen@gmail.com

> Issue #7549 has been updated by brixen (Brian Ford).

I've written another post addressing many of the misunderstandings about
my proposal expressed in this thread:

http://brixen.io/2012/12/30/a-ruby-design-process-talking-points

We are starting work on the Consensus application here:

https://github.com/rubyspec/consensus

Cheers,
Brian

Feature #7549: A Ruby Design Process
https://bugs.ruby-lang.org/issues/7549#change-35171

Author: brixen (Brian Ford)
Status: Open
Priority: Normal
Assignee:
Category:
Target version:

Matz,

At RubyConf 2012, I gave a talk about a design process for Ruby (
http://www.confreaks.com/videos/1278-rubyconf2012-toward-a-design-for-ruby).
So far, over 12,000 people have viewed that talk. I think it is reasonable
to say that many people are concerned about and interested in a design
process for Ruby.

On Monday, we had an IRC meeting of Ruby implementers. Most of the points
in my proposal were discussed but I'm concerned that a lot of confusion
remains.

I have written a post that describes a Ruby design process and hopefully
clarifies points that people found confusing:

http://brixen.io/2012/12/11/a-ruby-design-process

I would like to propose this process for making changes to Ruby. I am
going to put a summary of the process at http://rubyspec.org/design and
ask for people who support the process to submit their signature. I'd like
to request that you consider the response from the community for my
proposal.

Thank you,
Brian

--
http://bugs.ruby-lang.org/

--
NARUSE, Yui naruse@airemix.jp

#34 - 09/26/2014 09:56 AM - duerst (Martin Dürst)

- Status changed from Open to Rejected

I just read a blog that said this issue was still open.
I might be slightly biased (see my comments above), but in order to avoid confusion, I'm closing this issue because the proposal is no longer actively discussed.
If somebody has a better proposal, they are always welcome to open another issue.

#35 - 12/23/2021 11:40 PM - hsbt (Hiroshi SHIBATA)

- Project changed from 14 to Ruby