

Ruby master - Feature #16378

Support leading arguments together with ...

11/27/2019 04:50 PM - Eregon (Benoit Daloze)

Status:	Closed	
Priority:	Normal	
Assignee:	matz (Yukihiro Matsumoto)	
Target version:	3.0	
Description		
I think this is very important, otherwise ... can be used only very rarely.		
For instance, method_missing typically want to access the method name like:		
<pre>def method_missing(name, ...) if name.to_s.end_with?('?') self[name] else fallback(name, ...) end end</pre>		
See the original feature: https://bugs.ruby-lang.org/issues/16253#note-19 . I think most people expect def method_missing(name, ...) to work.		
Related issues:		
Related to Ruby master - Feature #16253: Shorthand "forward everything" syntax		Closed
Related to Ruby master - Feature #16891: Restore Positional Argument to Keywo...		Rejected
Related to Ruby master - Feature #16463: Fixing *args-delegation in Ruby 2.7:...		Closed

Associated revisions

Revision f8b4340f - 06/07/2020 12:07 AM - jeremyevans (Jeremy Evans)

Add leading arguments support to arguments forwarding

The idFWD_KWREST sections may be wrong. However, the existing idFWD_KWREST sections for ... without leading arguments are already broken.

Implements [Feature #16378]

Revision 27fca662 - 12/31/2020 06:19 AM - nagachika (Tomoyuki Chikanaga)

Add leading arguments support to arguments forwarding

Implements [Feature #16378]

History

#1 - 11/27/2019 04:50 PM - Eregon (Benoit Daloze)

- Related to Feature #16253: Shorthand "forward everything" syntax added

#2 - 11/27/2019 04:53 PM - Eregon (Benoit Daloze)

- Target version set to 2.7

- Description updated

#3 - 11/27/2019 04:57 PM - Eregon (Benoit Daloze)

Also I believe ... could be a good way to do delegation in all versions for lexical cases (the majority):

```
# Could be some constant in a gem
ARGS = RUBY_VERSION < "2.7" ? "*args, &block" : "..."
```

```
class_eval <<RUBY
def method_missing(name, #{ARGS})
```

```
if name.to_s.end_with?('?')
  self[name]
else
  fallback(name, #{ARGS})
end
end
RUBY
```

And while Redmine doesn't syntax highlight <<RUBY, at least GitHub and RubyMine do.

#4 - 11/28/2019 04:56 AM - matz (Yukihiko Matsumoto)

- Status changed from Open to Rejected

I know trailing ... can be very useful from C experience. But the primary purpose of Ruby ... is method delegation. We are not going to extend the role of ... in the language (at least for now).

Matz.

#5 - 11/28/2019 10:30 AM - Eregon (Benoit Daloze)

matz (Yukihiko Matsumoto) wrote:

I know trailing ... can be very useful from C experience. But the primary purpose of Ruby ... is method delegation. We are not going to extend the role of ... in the language (at least for now).

That is surprising.

It makes ... unusable in many delegation use cases which need to extract the first(s) arguments.

The above method_missing is also delegation, isn't it?

What's your solution for that case?

Using ruby2_keywords def method_missing(name, *args) and then having to change it to def method_missing(name, *args, **kwargs) once ruby2_keywords is removed?

Defining method_missing is not something rare in Ruby. It seems a shame ... can't be used there, even though it would a very good place to use ... (delegation in method_missing is almost always lexical).

BTW, R has ... and it supports leading arguments.

And of course the construct that ... replaces, that is *args, &block as "all arguments" supports leading arguments too.

#6 - 12/27/2019 06:15 PM - Eregon (Benoit Daloze)

- Target version changed from 2.7 to 3.0

- Assignee set to matz (Yukihiko Matsumoto)

- Status changed from Rejected to Open

[matz \(Yukihiko Matsumoto\)](#) Could you reply to this?

Particularly:

But the primary purpose of Ruby ... is method delegation.

Indeed, and I believe we also want to extract leading arguments in many delegation use cases.

... not supporting leading arguments is a obvious limitation and I would think unexpected for many rubyists.

As an example, it makes ... unusable for method_missing, which is a place where delegation often happens.

I think the decision was too quick, maybe because I set target version 2.7.

It won't be in 2.7 since that's released, but let's consider it for future releases.

#7 - 12/27/2019 07:54 PM - Dan0042 (Daniel DeLorme)

In the [DevelopersMeeting20191017Japan log](#) there was "Future work: lead argument handling is postponed", so clearly there was the intention of adding it later.

#8 - 02/27/2020 07:08 AM - matz (Yukihiko Matsumoto)

We have found out that #method_missing (and #send) needed leading arguments otherwise we cannot use argument forwarding for them. I changed my mind. Accepted.

Matz.

#9 - 05/28/2020 11:41 AM - Eregon (Benoit Daloze)

- Related to Feature #16891: Restore Positional Argument to Keyword Conversion added

#10 - 05/28/2020 11:44 AM - Eregon (Benoit Daloze)

Interesting, I never saw the reply above from matz.

Looking at my email I received one but it wasn't attached to the rest of the conversation for this issue for some reason.

Great to hear it's now accepted.

I think we should have it already in 2.7.2+.

Many people clearly want it in [#16891](#).

[jeremyevans0 \(Jeremy Evans\)](#) or [nobu \(Nobuyoshi Nakada\)](#) Could you implement it on master?

#11 - 05/28/2020 12:03 PM - Eregon (Benoit Daloze)

- Related to Feature #16463: Fixing *args-delegation in Ruby 2.7: ruby2_keywords semantics by default in 2.7.1 added

#12 - 05/28/2020 06:11 PM - jeremyevans0 (Jeremy Evans)

Eregon (Benoit Daloze) wrote in [#note-10](#):

[jeremyevans0 \(Jeremy Evans\)](#) or [nobu \(Nobuyoshi Nakada\)](#) Could you implement it on master?

I've implemented basic support, which passes make check:

<https://github.com/jeremyevans/ruby/commit/672901facc6f88ae40b4ea8ef59ef04efd5a5de>

I think ripper support is probably broken for it, and the idFWD_KWREST sections (not currently enabled) are probably also wrong.

[nobu \(Nobuyoshi Nakada\)](#), could you please fix the ripper support and idFWD_KWREST sections?

#13 - 06/05/2020 10:45 PM - jeremyevans0 (Jeremy Evans)

jeremyevans0 (Jeremy Evans) wrote in [#note-12](#):

Eregon (Benoit Daloze) wrote in [#note-10](#):

[jeremyevans0 \(Jeremy Evans\)](#) or [nobu \(Nobuyoshi Nakada\)](#) Could you implement it on master?

I've implemented basic support, which passes make check:

<https://github.com/jeremyevans/ruby/commit/672901facc6f88ae40b4ea8ef59ef04efd5a5de>

I think ripper support is probably broken for it, and the idFWD_KWREST sections (not currently enabled) are probably also wrong.

[nobu \(Nobuyoshi Nakada\)](#), could you please fix the ripper support and idFWD_KWREST sections?

From my testing, the existing idFWD_KWREST sections for ... without leading arguments are already broken. Defining RUBY3_KEYWORDS in parse.y results in broken code such as args being dropped and no implicit conversion of nil into Hash (TypeError) when using super. So I don't think there is a point trying to get idFWD_KWREST sections working for leading arguments currently. We are not going to want to switch to idFWD_KWREST sections until we drop ruby2_keywords support, as ruby2_keywords is significantly faster.

I updated the commit to add ripper support, and added a pull request for it: <https://github.com/ruby/ruby/pull/3190>

#14 - 06/07/2020 12:07 AM - jeremyevans (Jeremy Evans)

- Status changed from Open to Closed

Applied in changeset [git!f8b4340fa2c254cd093ebc3bc70d2d0c46ea9997](https://github.com/ruby/ruby/pull/3190).

Add leading arguments support to arguments forwarding

The idFWD_KWREST sections may be wrong. However, the existing idFWD_KWREST sections for ... without leading arguments are already broken.

Implements [Feature [#16378](#)]

#15 - 06/19/2020 07:40 AM - mame (Yusuke Endoh)

[nagachika \(Tomoyuki Chikanaga\)](#) Can we backport this to 2.7.2? Strictly speaking, this is a new feature, but according to [public consultation about Ruby 3.0 keyword change](#), this seems important to mitigate the pain of the change in 2.7. Matz also agreed with the backport.

[jeremyevans0 \(Jeremy Evans\)](#) If nagachika-san agreed with the backport, can you create a patch for backport?

#16 - 06/19/2020 04:22 PM - jeremyevans0 (Jeremy Evans)

- File `ruby_2_7_lead_args_forwarding.patch` added

mame (Yusuke Endoh) wrote in [#note-15](#):

[jeremyevans0 \(Jeremy Evans\)](#) If nagachika-san agreed with the backport, can you create a patch for backport?

Backport patch attached, in case [nagachika \(Tomoyuki Chikanaga\)](#) approves.

#17 - 06/29/2020 07:59 AM - nagachika (Tomoyuki Chikanaga)

Thank you for your notice and providing the patch. I will take a look into it.

#18 - 12/27/2020 07:41 PM - Dan0042 (Daniel DeLorme)

[nagachika \(Tomoyuki Chikanaga\)](#) I would really like to see this backported to 2.7 ... may I ask for the status?

#19 - 12/28/2020 12:35 PM - Eregon (Benoit Daloze)

Agreed it should be in 2.7 (well, I've said since the beginning of this issue :D).

Right now, I consider ... non-existing because it's almost never usable due to the restrictions in 2.7.

#20 - 12/31/2020 06:32 AM - nagachika (Tomoyuki Chikanaga)

Thank you for ping me. I backported f8b4340f into ruby_2_7 at 27fca66207f2c35f2f44f6a7cbbe6fd153546082.

Files

<code>ruby_2_7_lead_args_forwarding.patch</code>	8.98 KB	06/19/2020	jeremyevans0 (Jeremy Evans)
--	---------	------------	---