Ruby trunk - Bug #13196

Improve keyword argument errors when non-keyword arguments given

02/05/2017 08:18 PM - olivierlacan (Olivier Lacan)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:		**Backport:**	2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: UNKNOWN

Description

Given the following method definition:

```
def explode(code:)
  puts "Boom!"
end
```

If a Ruby user doesn't provide any arguments when calling the explode method, the following helpful feedback is given:

```
explode
ArgumentError: missing keyword: code
```

But when a Ruby user mistakenly provides a regular argument, the exception message is obtuse and unhelpful:

```
explode "1234"
ArgumentError: wrong number of arguments (given 1, expected 0)
```

This does not provide information to properly recover from the error. Worse, it's incorrect. It is not true that the method expected 0 arguments. The method expected 1 keyword argument.

Instead, Ruby should respond something like:

```
explode "1234"
ArgumentError: missing keyword: code, given "1234" which is not a keyword argument.
```

One could argue that this situation would call for a different error class, perhaps a KeywordArgumentError that would inherit from ArgumentError, but that would extend the scope of this feature request a bit too far in my mind.

Related issues:

Related to Ruby trunk - Bug #14176: Unclear error message when calling method...	**Open**

Associated revisions

Revision 1c34f0b8 - 07/04/2017 05:42 AM - nobu (Nobuyoshi Nakada)

vm_args.c: improve keyword argument errors

- vm_args.c (argument_arity_error): improve required keyword argument errors when non-keyword arguments given. [ruby-core:79439] [Bug #13196]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@59259 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 59259 - 07/04/2017 05:42 AM - nobu (Nobuyoshi Nakada)

vm_args.c: improve keyword argument errors

- vm_args.c (argument_arity_error): improve required keyword argument errors when non-keyword arguments given. [ruby-core:79439] [Bug #13196]

Revision 59259 - 07/04/2017 05:42 AM - nobu (Nobuyoshi Nakada)

vm_args.c: improve keyword argument errors

- vm_args.c (argument_arity_error): improve required keyword argument errors when non-keyword arguments given. [ruby-core:79439] [Bug #13196]

Revision 59259 - 07/04/2017 05:42 AM - nobu (Nobuyoshi Nakada)

vm_args.c: improve keyword argument errors

- vm_args.c (argument_arity_error): improve required keyword argument errors when non-keyword arguments given. [ruby-core:79439] [Bug #13196]

History

#1 - 02/06/2017 03:24 AM - shevegen (Robert A. Heiler)

I guess the error-reporting there did not yet account for the possibility that keyword arguments can be mandatory too, so I agree that the message "wrong number of arguments (given 1, expected 0)" appears to be incorrect, since one indeed has to pass a mandatory argument to that method. I assume that when keyword args were added, this behaviour was probably not yet noticed. I have not seen code like "foo:" in any method definition yet either, though.

#2 - 02/06/2017 04:09 AM - nobu (Nobuyoshi Nakada)

How about this?

```
$ ./miniruby -e 'def explode(code:)end' -e 'explode(1)'
-e:1:in `explode': wrong number of arguments (given 1, expected 0 with required keyword code) (ArgumentError)
    from -e:2:in `<main>'
```

```
diff --git a/vm_args.c b/vm_args.c
index 6cded80924..76516f60e0 100644
--- a/vm_args.c
+++ b/vm_args.c
@@ -725,7 +725,25 @@ raise_argument_error(rb_thread_t *th, const rb_iseq_t *iseq, const VALUE exc)
 static void
 argument_arity_error(rb_thread_t *th, const rb_iseq_t *iseq, const int miss_argc, const int min_argc, const i
nt max_argc)
 {
-    raise_argument_error(th, iseq, rb_arity_error_new(miss_argc, min_argc, max_argc));
+    VALUE exc = rb_arity_error_new(miss_argc, min_argc, max_argc);
+    if (iseq->body->param.flags.has_kw) {
+   const struct rb_iseq_param_keyword *const kw = iseq->body->param.keyword;
+   const ID *keywords = kw->table;
+   int req_key_num = kw->required_num;
+   if (req_key_num > 0) {
+       VALUE mesg = rb_attr_get(exc, idMesg);
+       rb_str_resize(mesg, RSTRING_LEN(mesg)-1);
+       rb_str_cat_cstr(mesg, " with required keyword");
+       if (req_key_num > 1) rb_str_cat_cstr(mesg, "s");
+       do {
+       rb_str_cat_cstr(mesg, " ");
+       rb_str_append(mesg, rb_id2str(*keywords++));
+       rb_str_cat_cstr(mesg, ",");
+       } while (--req_key_num);
+       RSTRING_PTR(mesg)[RSTRING_LEN(mesg)-1] = ')';
+   }
+    }
+    raise_argument_error(th, iseq, exc);
 }

 static void
```

#3 - 02/09/2017 08:00 PM - stomar (Marcus Stollsteimer)

Nobuyoshi Nakada wrote:

> wrong number of arguments (given 1, expected 0 with required keyword code)

IMO still unclear, sounds somewhat like "given 1 argument with required keyword code, but expected 0". Also, code would have to be quoted (keyword `code') or otherwise marked as an identifier, or it could easily be interpreted as normal part of the message.

Maybe something like this(?):

```
wrong number of arguments (given 1, expected 0; missing keywords: code, foo)
```

which would only require a slight change in your patch.

#4 - 02/26/2017 03:26 AM - olivierlacan (Olivier Lacan)

- File missing_kwargs.diff added

Marcus Stollsteimer wrote:

> wrong number of arguments (given 1, expected 0; missing keywords: code, foo)

I agree that this is clearer so I slightly modified Nobu's patch:

```
diff --git a/vm_args.c b/vm_args.c
index 6cded80924..76516f60e0 100644
--- a/vm_args.c
+++ b/vm_args.c
@@ -725,7 +725,25 @@ raise_argument_error(rb_thread_t *th, const rb_iseq_t *iseq, const VALUE exc)
 static void
 argument_arity_error(rb_thread_t *th, const rb_iseq_t *iseq, const int miss_argc, const int min_argc, const i
nt max_argc)
 {
-    raise_argument_error(th, iseq, rb_arity_error_new(miss_argc, min_argc, max_argc));
+    VALUE exc = rb_arity_error_new(miss_argc, min_argc, max_argc);
+    if (iseq->body->param.flags.has_kw) {
+   const struct rb_iseq_param_keyword *const kw = iseq->body->param.keyword;
+   const ID *keywords = kw->table;
+   int req_key_num = kw->required_num;
+   if (req_key_num > 0) {
+       VALUE mesg = rb_attr_get(exc, idMesg);
+       rb_str_resize(mesg, RSTRING_LEN(mesg)-1);
+       rb_str_cat_cstr(mesg, "; required keyword");
+       if (req_key_num > 1) rb_str_cat_cstr(mesg, "s");
+       do {
+       rb_str_cat_cstr(mesg, ": ");
+       rb_str_append(mesg, rb_id2str(*keywords++));
+       rb_str_cat_cstr(mesg, ",");
+       } while (--req_key_num);
+       RSTRING_PTR(mesg)[RSTRING_LEN(mesg)-1] = ')';
+   }
+    }
+    raise_argument_error(th, iseq, exc);
 }

 static void
```

#5 - 02/26/2017 03:40 AM - olivierlacan (Olivier Lacan)

Although I find Nobu's patch excellent, it still bothers me that the exception says expected 0.

Keyword arguments do increase a method's arity just like regular arguments, so the message is both confusing *and* disingenuous.

Given the following definition:

```
def explode(code:)
  puts "Boom!"
end
```

And the following call:

```
explode "1234"
```

I would much rather receive the following exception message:

```
ArgumentError: invalid argument "1234" (expected keyword arguments: :code, :foo)
```

First observation, since the keywords are symbols, they should be referred to as symbols — otherwise users may incorrectly try to pass variables named code and foo to recover from the error. Yes, I know it's silly but that appears to be what the exception message suggests if code and foo are referenced without colons.

Second observation, while it could be unwieldy to display the submitted arguments (especially if there are many and their representation is long), it also makes for a much clearer context for the feedback being given.

An alternative would be:

```
ArgumentError: 1 unexpected non-keyword argument (expected keyword arguments: :code, :foo)
```

#6 - 02/26/2017 09:07 PM - stomar (Marcus Stollsteimer)

Olivier Lacan wrote:

```
+   if (req_key_num > 0) {
+       VALUE mesg = rb_attr_get(exc, idMesg);
+       rb_str_resize(mesg, RSTRING_LEN(mesg)-1);
+       rb_str_cat_cstr(mesg, "; required keyword");
+       if (req_key_num > 1) rb_str_cat_cstr(mesg, "s");
+       do {
+       rb_str_cat_cstr(mesg, ": ");
+       rb_str_append(mesg, rb_id2str(*keywords++));
+       rb_str_cat_cstr(mesg, ",");
+       } while (--req_key_num);
```

- I think rb_str_cat_cstr(mesg, ": "); in the while-loop doesn't work; ":" must be added before the loop, in the loop only " " to separate different entries
- suggestion: s/keyword/keyword argument/, since "keyword" might be confused with language keywords like def, end, ...

Regarding code vs. :code, IMHO for beginners it's much easier to understand without leading ":", similar to the usage in the call sequence, it's not explode(:code: 123) but explode(code: 123), and in the method body.

#7 - 02/27/2017 01:50 AM - duerst (Martin Dürst)

Marcus Stollsteimer wrote:

> Regarding code vs. :code, IMHO for beginners it's much easier to understand without leading ":", similar to the usage in the call sequence, it's not explode(:code: 123) but explode(code: 123), and in the method body.

Yes, actually, if a colon is needed at all, I'd put it at the end of the keyword(s), because that's how it appears in the method invocation:

```
ArgumentError: 1 unexpected non-keyword argument (expected keyword arguments: code:, foo:)
```

But I think the colons are unnecessary; the name of the arguments is code and foo; that these names are expressed as symbols in *some* contexts and that symbols are denoted with colons before or after are syntactic details depending on the context.

#8 - 03/16/2017 08:10 AM - olivierlacan (Olivier Lacan)

- File missing_kwargs.diff added

stomar (Marcus Stollsteimer) wrote:

> - I think rb_str_cat_cstr(mesg, ": "); in the while-loop doesn't work; ":" must be added before the loop, in the loop only " " to separate different entries

Woops, good catch. Fixed in a new attached patch.

> - suggestion: s/keyword/keyword argument/, since "keyword" might be confused with language keywords like def, end, ...

Can't do that since the normal kwargs error is:

> ArgumentError: missing keywords: code, token

> Regarding code vs. :code, IMHO for beginners it's much easier to understand without leading ":", similar to the usage in the call sequence, it's not explode(:code: 123) but explode(code: 123), and in the method body.
> I've changed my mind on this and I agree. Especially since the above existing error lists references the keyword arguments without colons.

duerst (Martin Dürst) wrote:

> Yes, actually, if a colon is needed at all, I'd put it at the end of the keyword(s), because that's how it appears in the method invocation:

Looks a bit odd, doesn't it? I could be convinced but this is beyond the scope of this patch and issue since there's existing error messages using no colons at all.

Here's the patch tested on trunk (2.5.0 dev):

```
irb(main):001:0> def explode(code:,token:)
irb(main):002:1> puts "Boom!"
irb(main):003:1> end
```

```
=> :explode
irb(main):004:0> explode
ArgumentError: missing keywords: code, token
    from (irb):1:in `explode'
    from (irb):4
    from /usr/local/bin/irb:11:in `<main>'
irb(main):005:0> explode "1234"
ArgumentError: wrong number of arguments (given 1, expected 0; required keywords: code, token)
    from (irb):1:in `explode'
    from (irb):5
    from /usr/local/bin/irb:11:in `<main>'
irb(main):006:0> RUBY_VERSION
=> "2.5.0"
```

#9 - 05/19/2017 06:07 AM - matz (Yukihiro Matsumoto)

Agreed with better message description.

Matz.

#10 - 07/04/2017 12:31 AM - olivierlacan (Olivier Lacan)

matz (Yukihiro Matsumoto) wrote:

> Agreed with better message description.

Thank you. Is there anything I can do to help move this issue along? Is the patch sufficient?

#11 - 07/04/2017 05:42 AM - nobu (Nobuyoshi Nakada)

- Status changed from Open to Closed

Applied in changeset trunk|r59259.

---

vm_args.c: improve keyword argument errors

- vm_args.c (argument_arity_error): improve required keyword argument errors when non-keyword arguments given. [ruby-core:79439] [Bug #13196]

#12 - 12/13/2017 06:34 PM - marcandre (Marc-Andre Lafortune)

- Related to Bug #14176: Unclear error message when calling method with keyword arguments added

Files

missing_kwargs.diff	1.2 KB	02/26/2017	olivierlacan (Olivier Lacan)
missing_kwargs.diff	1.23 KB	03/16/2017	olivierlacan (Olivier Lacan)