Backport200 - Backport #8040

Unexpect behavior when using keyword arguments

03/08/2013 03:39 AM - pabloh (Pablo Herrero)

Status:	Closed	
Priority:	Normal	
Assignee:	nagachika (Tomoyuki Chikanaga)	

Description

=begin
There is an odd behavior when calling methods with the new keyword arguments syntax, when you have a method defined with mandatory arguments that also takes options, like this:

```
def foo value, **keywords
puts [value,keywords].inspect
end

foo("somthing") #This works
foo("somthing", key: 'value') #This also works

foo(Hash.new(something: 'else')) #This raises 'ArgumentError: wrong number of arguments (0 for 1)'
```

This feels weird regardless the fact that keyword arguments are a Hash at the bottom, since you ARE PASSING an argument.

Other side effect from this, is that when you call the method ((|foo|)) with a single argument, you can't anticipate how many argument you will be actually passing at runtime unless you know beforehand the argument's class.

What's maybe even more concerning is the fact than this happens even when you pass an argument which class derives from Hash:

```
class MyDirectory < Hash; end

foo(MyDirectory.new(something: 'else')) #This also raises 'ArgumentError: wrong number of arguments (0 for 1)'
```

Besides finding this behavior surprising, I think this could also possibly lead to old code been unexpectedly broken when updating old methods that takes options to the new syntax.

For example if you have this code:

```
def foo_with_options argument, options = {}
#Do some stuff with options
end

#And at someplace else...

my_hash_thingy = Hash.new
foo_with_options(my_hash_thingy) #Only passing mandatory argument, with no options, works fine.
```

When updating to the new syntax:

```
def foo_with_options argument, an_option: 'value1', another_option: 'value2'
#Do some stuff with options
end

#And at someplace else...

my_hash_thingy = Hash.new
foo_with_options(my_hash_thingy) #Only passing mandatory argument, with no options, doesn't work anymore.
```

=end

Related issues:

Related to Ruby master - Feature #14183: "Real" keyword argument	**Closed**
Has duplicate Ruby master - Bug #8316: Can't pass hash to first positional ar...	**Closed**

Associated revisions

Revision 6f9f8d2e - 05/30/2013 10:50 AM - mame (Yusuke Endoh)

- vm_insnhelper.c (vm_callee_setup_keyword_arg,
  vm_callee_setup_arg_complex): consider a hash argument for keyword
  only when the number of arguments is more than the expected
  mandatory parameters. [ruby-core:53199] [ruby-trunk - Bug #8040]

- test/ruby/test_keyword.rb: update a test for above.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@40992 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 40992 - 05/30/2013 10:50 AM - mame (Yusuke Endoh)

- vm_insnhelper.c (vm_callee_setup_keyword_arg,
  vm_callee_setup_arg_complex): consider a hash argument for keyword
  only when the number of arguments is more than the expected
  mandatory parameters. [ruby-core:53199] [ruby-trunk - Bug #8040]

- test/ruby/test_keyword.rb: update a test for above.

Revision 40992 - 05/30/2013 10:50 AM - mame (Yusuke Endoh)

- vm_insnhelper.c (vm_callee_setup_keyword_arg,
  vm_callee_setup_arg_complex): consider a hash argument for keyword
  only when the number of arguments is more than the expected
  mandatory parameters. [ruby-core:53199] [ruby-trunk - Bug #8040]

- test/ruby/test_keyword.rb: update a test for above.

Revision 40992 - 05/30/2013 10:50 AM - mame (Yusuke Endoh)

- vm_insnhelper.c (vm_callee_setup_keyword_arg,
  vm_callee_setup_arg_complex): consider a hash argument for keyword
  only when the number of arguments is more than the expected
  mandatory parameters. [ruby-core:53199] [ruby-trunk - Bug #8040]

- test/ruby/test_keyword.rb: update a test for above.

Revision 40992 - 05/30/2013 10:50 AM - mame (Yusuke Endoh)

- vm_insnhelper.c (vm_callee_setup_keyword_arg,
  vm_callee_setup_arg_complex): consider a hash argument for keyword
  only when the number of arguments is more than the expected
  mandatory parameters. [ruby-core:53199] [ruby-trunk - Bug #8040]

- test/ruby/test_keyword.rb: update a test for above.

Revision 40992 - 05/30/2013 10:50 AM - mame (Yusuke Endoh)

- vm_insnhelper.c (vm_callee_setup_keyword_arg,
  vm_callee_setup_arg_complex): consider a hash argument for keyword
  only when the number of arguments is more than the expected
  mandatory parameters. [ruby-core:53199] [ruby-trunk - Bug #8040]

- test/ruby/test_keyword.rb: update a test for above.

Revision 40992 - 05/30/2013 10:50 AM - mame (Yusuke Endoh)

- vm_insnhelper.c (vm_callee_setup_keyword_arg,
  vm_callee_setup_arg_complex): consider a hash argument for keyword
  only when the number of arguments is more than the expected
  mandatory parameters. [ruby-core:53199] [ruby-trunk - Bug #8040]

- test/ruby/test_keyword.rb: update a test for above.

Revision 2392c12b - 06/04/2013 02:30 PM - nagachika (Tomoyuki Chikanaga)

merge revision(s) 40992: [Backport #8040]

```
    * vm_insnhelper.c (vm_callee_setup_keyword_arg,
      vm_callee_setup_arg_complex): consider a hash argument for keyword
      only when the number of arguments is more than the expected
      mandatory parameters.  [ruby-core:53199] [ruby-trunk - Bug #8040]

    * test/ruby/test_keyword.rb: update a test for above.
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_0_0@41063 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 41063 - 06/04/2013 02:30 PM - nagachika (Tomoyuki Chikanaga)

merge revision(s) 40992: [Backport #8040]

```
* vm_insnhelper.c (vm_callee_setup_keyword_arg,
  vm_callee_setup_arg_complex): consider a hash argument for keyword
  only when the number of arguments is more than the expected
  mandatory parameters.  [ruby-core:53199] [ruby-trunk - Bug #8040]

* test/ruby/test_keyword.rb: update a test for above.
```

History

#1 - 03/10/2013 05:30 PM - mame (Yusuke Endoh)

- Status changed from Open to Assigned

- Assignee set to matz (Yukihiro Matsumoto)

This is a duplicate of #7529 which was once rejected by matz. I translate his reason:

> Unfortunately, it is the spec; we cannot distinguish whether the hash is for a keyword or for just an argument.
> Please add {} at the last.

But, it may be good that we consider the hash for a keyword *only when* the number of arguments is more than the expected mandatory parameters.

diff --git a/vm_insnhelper.c b/vm_insnhelper.c
index 0c447aa..2432288 100644
--- a/vm_insnhelper.c
+++ b/vm_insnhelper.c
@@ -1065,12 +1065,12 @@ vm_caller_setup_args(const rb_thread_t *th, rb_control_frame_t *cfp, rb_call_inf
}

static inline int
-vm_callee_setup_keyword_arg(const rb_iseq_t *iseq, int argc, VALUE *orig_argv, VALUE *kwd)
+vm_callee_setup_keyword_arg(const rb_iseq_t *iseq, int argc, int m, VALUE *orig_argv, VALUE *kwd)
{

VALUE keyword_hash;
int i, j;

- if (argc > 0 &&

- if (argc > m &&
  !NIL_P(keyword_hash = rb_check_hash_type(orig_argv[argc-1]))) {
  argc--;
  keyword_hash = rb_hash_dup(keyword_hash);
  @@ -1109,7 +1109,7 @@ vm_callee_setup_arg_complex(rb_thread_t *th, rb_call_info_t *ci, const rb_iseq_t

  /* keyword argument */
  if (iseq->arg_keyword != -1) {

- argc = vm_callee_setup_keyword_arg(iseq, argc, orig_argv, &keyword_hash);

- argc = vm_callee_setup_keyword_arg(iseq, argc, m, orig_argv, &keyword_hash);
  }

  /* mandatory */
  @@ -2127,7 +2127,7 @@ vm_yield_setup_block_args(rb_thread_t *th, const rb_iseq_t * iseq,

  /* keyword argument */
  if (iseq->arg_keyword != -1) {

- argc = vm_callee_setup_keyword_arg(iseq, argc, argv, &keyword_hash);

- argc = vm_callee_setup_keyword_arg(iseq, argc, m, argv, &keyword_hash);
  }

  /*

--
Yusuke Endoh mame@tsg.ne.jp

#2 - 03/11/2013 10:41 AM - marcandre (Marc-Andre Lafortune)

mame (Yusuke Endoh) wrote:

> But, it may be good that we consider the hash for a keyword *only when* the number of arguments is more than the expected mandatory parameters.

+1

#3 - 03/18/2013 11:36 AM - pabloh (Pablo Herrero)

I also like Yusuke's approach. Is it been considered?

#4 - 03/19/2013 03:23 AM - Anonymous

Matz: could you please confirm that mandatory arguments should be fulfilled
before checking for hash argument to fulfill named parameters?

BTW, also asked on StackOverflow today:

http://stackoverflow.com/questions/15480236/how-can-i-prevent-a-positional-argument-from-being-expanded-into-keyword-argumen
http://stackoverflow.com/questions/15480236/how-can-i-prevent-a-positional-argument-from-being-expanded-into-keyword-argumen/15483828#15483828

On Sun, Mar 17, 2013 at 10:36 PM, pabloh (Pablo Herrero) <
pablodherrero@gmail.com> wrote:

> Issue #8040 has been updated by pabloh (Pablo Herrero).
>
> I also like Yusuke's approach. Is it been considered?
>
> ---
>
> Bug #8040: Unexpect behavior when using keyword arguments
> https://bugs.ruby-lang.org/issues/8040#change-37685
>
> Author: pabloh (Pablo Herrero)
> Status: Assigned

Priority: Normal
Assignee: matz (Yukihiro Matsumoto)
Category:
Target version:
ruby -v: 2.0.0-p0

=begin
There is an odd behavior when calling methods with the new keyword
arguments syntax, when you have a method defined with mandatory arguments
that also takes options, like this:

def foo value, **keywords
puts [value,keywords].inspect
end

foo("somthing") #This works
foo("somthing", key: 'value') #This also works

foo(Hash.new(something: 'else')) #This raises 'ArgumentError: wrong
number of arguments (0 for 1)'

This feels weird regardless the fact that keyword arguments are a Hash at
the bottom, since you ARE PASSING an argument.

Other side effect from this, is that when you call the method ((|foo|))
with a single argument, you can't anticipate how many argument you will be
actually passing at runtime unless you know beforehand the argument's class.

What's maybe even more concerning is the fact than this happens even when
you pass an argument which class derives from Hash:

class MyDirectory < Hash; end

foo(MyDirectory.new(something: 'else')) #This also raises
'ArgumentError: wrong number of arguments (0 for 1)'

Besides finding this behavior surprising, I think this could also possibly
lead to old code been unexpectedly broken when updating old methods that
takes options to the new syntax.

For example if you have this code:

def foo_with_options argument, options = {}
#Do some stuff with options
end

#And at someplace else...

my_hash_thingy = Hash.new
foo_with_options(my_hash_thingy) #Only passing mandatory argument, with
no options, works fine.

When updating to the new syntax:

def foo_with_options argument, an_option: 'value1', another_option:
'value2'
#Do some stuff with options
end

#And at someplace else...

my_hash_thingy = Hash.new
foo_with_options(my_hash_thingy) #Only passing mandatory argument, with
no options, doesn't work anymore.
=end

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

#5 - 03/19/2013 10:49 AM - matz (Yukihiro Matsumoto)

- Assignee changed from matz (Yukihiro Matsumoto) to mame (Yusuke Endoh)

Accepted. Prioritize mandatory argument sounds reasonable.

Matz.

#6 - 04/10/2013 02:32 AM - pabloh (Pablo Herrero)

Any news about this?

#7 - 04/25/2013 11:52 AM - marcandre (Marc-Andre Lafortune)

Any hope of getting this in time for next patchlevel release?

#8 - 04/26/2013 03:53 PM - zzak (Zachary Scott)

Assign to nagachika-san

#9 - 05/06/2013 10:32 PM - nagachika (Tomoyuki Chikanaga)

ping?

#10 - 05/30/2013 07:50 PM - mame (Yusuke Endoh)

- Status changed from Assigned to Closed

- % Done changed from 0 to 100

This issue was solved with changeset r40992.
Pablo, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

- vm_insnhelper.c (vm_callee_setup_keyword_arg,
  vm_callee_setup_arg_complex): consider a hash argument for keyword
  only when the number of arguments is more than the expected
  mandatory parameters. [ruby-core:53199] [ruby-trunk - Bug #8040]

- test/ruby/test_keyword.rb: update a test for above.

#11 - 05/30/2013 07:52 PM - mame (Yusuke Endoh)

- Tracker changed from Bug to Backport

- Project changed from Ruby master to Backport200

- Status changed from Closed to Assigned

- Assignee changed from mame (Yusuke Endoh) to nagachika (Tomoyuki Chikanaga)

Fixed. Sorry for very very late action!
I left to nagachika-san whether it should be backported or not.

--
Yusuke Endoh mame@tsg.ne.jp

#12 - 05/30/2013 10:37 PM - nagachika (Tomoyuki Chikanaga)

- Priority changed from Normal to 5

I also think new behavior is better than former behavior.
I'll backport it till next release.

#13 - 06/04/2013 11:30 PM - nagachika (Tomoyuki Chikanaga)

- Status changed from Assigned to Closed

This issue was solved with changeset r41063.
Pablo, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

---

merge revision(s) 40992: [Backport #8040]

```
* vm_insnhelper.c (vm_callee_setup_keyword_arg,
  vm_callee_setup_arg_complex): consider a hash argument for keyword
  only when the number of arguments is more than the expected
  mandatory parameters.  [ruby-core:53199] [ruby-trunk - Bug #8040]

* test/ruby/test_keyword.rb: update a test for above.
```

#14 - 12/14/2017 07:23 AM - hsbt (Hiroshi SHIBATA)

- Related to Feature #14183: "Real" keyword argument added