Ruby trunk - Bug #12136

OpenStruct.new(format: :bar).send :format # => too few arguments

03/02/2016 11:19 AM - niko (Niko Dittmann)

Status:	Open		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 2.3.0p0 (2015-12-25 revision 53290) [x86_64-linux]	**Backport:**	2.1: UNKNOWN, 2.2: UNKNOWN, 2.3: UNKNOWN

Description

#send(:format) to an OpenStruct with a field named :format raises an ArgumentError in Ruby 2.3.0:

```
OpenStruct.new(format: :bar).send :format
ArgumentError: too few arguments
```

It works as expected in ruby 2.2.1p85 (2015-02-26 revision 49769) [x86_64-linux] and with any other method name I tried:

```
OpenStruct.new(f: :bar).send :f
=> :bar
```

String or Symbol in the OpenStruct definition and as argument of #send make no difference.

Related issues:

Related to Ruby trunk - Bug #12251: DelegateClass(OpenStruct) behavior in 2.3...	**Open**

History

#1 - 03/02/2016 12:53 PM - niko (Niko Dittmann)

It's this commit: https://github.com/ruby/ruby/blob/7fa21558051e5412dcb790f528e392476edd4389/lib/ostruct.rb

By defining the getters and setters lazily the Kernel, Object and BasicObject instance methods shine through and #method_missing doesn't kick in. Therefor the #send semantics is broken for methods colliding with methods defined in parent classes.

#2 - 03/04/2016 12:45 AM - marcandre (Marc-Andre Lafortune)

Indeed, latest optimization of OpenStruct now allows conflicts with Object private methods.

I didn't realize it, but conflicts with public methods are already ignored (i.e. OpenStruct.new(hash: 'code').hash does not return 'code)

Note that OpenStruct.new(format: :bar).public_send :format does return :bar.

Possibilities:

a) Keep behavior the same and rubyists can alleviate these by using public_send instead of send

b) Modify new to check for conflict between keys and Object private instance methods and define actual methods in these cases.

c) Revert optimization. Optionally create OpenStruct.lazy for the optimized version.

I'm in favor for the later, but maybe I'm missing alternatives?

BTW, I thought at first that we could undefine private instance methods of OpenStruct, except for the usual callbacks and modify respond_to_missing? + method_missing so that calls to these private methods still work.

Sadly, there's no way to know from method_missing if that method is called privately or publicly, so this would effectively make all private methods become public which is not acceptable.

#3 - 03/04/2016 09:57 AM - Eregon (Benoit Daloze)

Marc-Andre Lafortune wrote:

> Sadly, there's no way to know from method_missing if that method is called privately or publicly, so this would effectively make all private methods become public which is not acceptable.

There is a way now, mentioned in #12113. I'm not sure whether it is good idea to use it, though.

#4 - 04/06/2016 05:13 PM - marcandre (Marc-Andre Lafortune)

- Related to Bug #12251: DelegateClass(OpenStruct) behavior in 2.3.0 different from 2.2 added

#5 - 04/06/2016 06:26 PM - dblock (Daniel Doubrovkine)

> a) Keep behavior the same and rubyists can alleviate these by using public_send instead of send

It doesn't seem that swapping send by public_send has any effect, at least not in the example in #12251.

Is there a workaround for existing code that would make things work the way it worked in Ruby 2.2.x?

#6 - 05/29/2016 01:35 PM - marcandre (Marc-Andre Lafortune)

- Has duplicate Bug #12349: Can't load OpenStruct with Syck with Ruby 2.3.x added

#7 - 05/29/2016 01:36 PM - marcandre (Marc-Andre Lafortune)

- Has duplicate deleted (Bug #12349: Can't load OpenStruct with Syck with Ruby 2.3.x)