Ruby master - Bug #6722

Weird behavior of defined?(super) check invoked from a metaprogrammatically defined class method

07/11/2012 07:59 PM - alexisowl (Alexey Smolianinov)

| Status:    | Closed                  |
| Priority:  | Normal                  |
| Assignee:  |                         |
| Target version: | 2.0.0                  |
| Backport:  |                         |

ruby -v: ruby 1.9.3p0 (2011-10-30 revision 33570) [i686-linux]

Description

=begin
Hi guys,

Sorry for the duplicate - I already reported ((())) 2 weeks ago, but didn't receive any answer.
I had mistakenly assigned "target version", maybe that was the reason of no response? Please let me know if my bug report is wrong or irrelevant in any other respect.

Here's a link to the discussion of the issue on stackoverflow:
((URL:http://stackoverflow.com/questions/11180909))

Bellow is my initial description:

--
Lately I came across some weird behavior with the (({defined?})) operator used to check if (({super})) keyword can be invoked in current context. Usually it works fine, but when I tried to combine the (({defined? super})) check with a tiny bit of metaprogramming, it gave me unexpected results.

class A;
def self.def_f!;
   singleton_class.send(:define_method, :f) { defined? super }
end
end
class AA < A; end

A.def_f!
A.f # => nil
AA.f # => nil

AA.def_f!
AA.f # => "super"
A.f # => "super" - WHY???

The last result is really strange, isn't it?

(({(A.f)}) has no super method, so I would expect the last (((A.f))) to return (((nil))).
=end

Related issues:

Is duplicate of Ruby master - Bug #6644: Weird behavior of defined?(super) ch...

Closed 06/25/2012

History

#1 - 07/12/2012 05:11 AM - nobu (Nobuyoshi Nakada)

- Status changed from Open to Closed
This issue was solved with changeset r36369.
Alexey, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

defined: me in cfp

- insns.def (defined): use method entry and id in cfp for proper superclass, since klass in iseq is shared by dynamically defined methods from the same block. [ruby-core:45831][Bug #6644]

#2 - 07/12/2012 05:38 AM - nobu (Nobuyoshi Nakada)
- Category set to core
- Target version set to 2.0.0
- % Done changed from 0 to 100