Description
C_User#y is refined. Is that intentional?

class C
def foo
  p :C_foo
end
end

module RefineC
  refine C do
    def foo
      p :RefineC_foo
      super
    end
  end
end

class C_User
  using RefineC
def x
    C.new.foo
  end
end

class C_User
  def y
    C.new.foo
  end
end
C_User.new.x
C_User.new.y

#=>
:RefineC_foo
:C_foo
:RefineC_foo
:C_foo

Related issues:
Related to Ruby master - Feature #4085: Refinements and nested methods

History
#1 - 11/05/2012 11:57 AM - shugo (Shugo Maeda)

ko1 (Koichi Sasada) wrote:

  C_User#y is refined. Is that intentional?

Yes, it's intentional. Refinements are activated when C_User is reopens.
I'm not sure whether is a good idea or not, but if C_User.module_eval is affected by refinements, it seems natural reopened definitions of C_User are also affected.

#2 - 11/05/2012 03:59 PM - ko1 (Koichi Sasada)
(2012/11/05 11:57), shugo (Shugo Maeda) wrote:

Yes, it's intentional. Refinements are activated when C_User is reopend.

I'm not sure whether is a good idea or not, but if C_User.module_eval is affected by refinements, it seems natural reopened definitions of C_User are also affected.

I can't understand that refinement is "lexical" or not.

If it affected after re-open, then it is not "lexical".
If the location of 'using' affect the result, then it seems "lexical".

([ruby-core:48773] [ruby-trunk - Bug #7269])

--
// SASADA Koichi at dot dot net

#3 - 11/05/2012 04:38 PM - shugo (Shugo Maeda)
ko1 (Koichi Sasada) wrote:

(2012/11/05 11:57), shugo (Shugo Maeda) wrote:

Yes, it's intentional. Refinements are activated when C_User is reopend.

I'm not sure whether is a good idea or not, but if C_User.module_eval is affected by refinements, it seems natural reopened definitions of C_User are also affected.

I can't understand that refinement is "lexical" or not.

If it affected after re-open, then it is not "lexical".
If the location of 'using' affect the result, then it seems "lexical".

([ruby-core:48773] [ruby-trunk - Bug #7269])

I may have confused you by the word lexical, but I used the word lexical in the sense that refinements have no local rebinding. I'm not sure whether refinements should be literally lexical or not.

I guess if refinements are literally lexical it's good for performance, but how module_eval (or an alternative to apply refinements to a given block) should behave?

#4 - 12/13/2012 02:25 PM - shugo (Shugo Maeda)
- Status changed from Assigned to Closed

Refinements are not activated by reopen now, so I close this ticket.