Ruby trunk - Bug #14012

NameError is raised when use class variables in Refinements

10/13/2017 11:32 AM - joker1007 (Tomohiro Hashidate)

Status:	Open		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 2.4.2p198 (2017-09-14 revision 59899) [x86_64-linux]	**Backport:**	2.3: UNKNOWN, 2.4: UNKNOWN

Description

Sorry in advance if other ticket exists.

In a case, Reference to class variables raises unnatural NameError.

A class variable is defined in a module.
And include the module in "refine" block.
Refined method cannot use the class variable, and raises NameError.

Sample.

```
module FooMod
  @@foo_mod1 = 1

  def foo_mod1
    p self.class.class_variable_get("@@foo_mod1")
  end
end

module FooMod2
  @@foo_mod2 = 1

  def foo_mod2
    p self.class.class_variable_get("@@foo_mod2")
  end
end

class Foo
  @@hoge = 1

  include FooMod

  def hoge1
    p @@hoge
  end
end

module Ext
  refine Foo do
    def hoge2
      p self.class.class_variable_get("@@hoge")
    end

    include FooMod2
  end
end

using Ext

Foo.new.hoge1 # => 1 OK
Foo.new.hoge2 # => 1 OK
Foo.new.foo_mod1 # => 1 OK
Foo.new.foo_mod2 # => uninitialized class variable @@foo_mod2 in Foo (NameError)
```

Is This behavior Refinements spec? or bug?

Related issues:

Related to Ruby trunk - Feature #12533: Refinements: allow modules inclusion,...	**Assigned**

History

#1 - 10/13/2017 01:43 PM - shevegen (Robert A. Heiler)

I have no idea if this is a bug or a feature, but I believe it seems more
of a bug. The @@variables should be accessible within the namespace of
the specific class right? So it should be viewable in a refinement too -
but admittedly, I have no idea about the specification of refinements
and I also have not been using @@variables in ages either. I just think
that this is more likely to be a bug.

#2 - 10/13/2017 04:42 PM - nobu (Nobuyoshi Nakada)

- Related to Feature #12533: Refinements: allow modules inclusion, in which the module can call internal methods which it defines. added