Ruby - Bug #11246

refine block doesn't respect "lexical" refinement information

06/10/2015 04:24 PM - ko1 (Koichi Sasada)

Status:	Rejected		
Priority:	Normal		
Assignee:	shugo (Shugo Maeda)		
Target version:			
ruby -v:	ruby 2.3dev	**Backport:**	2.0.0: UNKNOWN, 2.1: UNKNOWN, 2.2: UNKNOWN

Description

The following program making two refinements refine class C.

```
class C
  def foo
    p C
  end
end

module R1
  refine C do
    def foo
      p R1
      super
    end
  end
end

using R1 # 1

module R2
  using R1 # 2

  refine C do
    # using R1 # 3

    def bar
      C.new.foo
    end
  end
end

using R2

C.new.bar
```

Without using R1 # 3, C#foo was called in R2::C#bar.

By using R1 #1 and #2, we declared that this lexical scope should use R1. However, it seems that this declaration is ignored.

Is it an intentional behavior?

History

#1 - 06/10/2015 04:25 PM - ko1 (Koichi Sasada)

- Description updated

#2 - 06/24/2015 06:02 AM - shugo (Shugo Maeda)

- Status changed from Open to Rejected

Koichi Sasada wrote:

> The following program making two refinements refine class C.
>
> ```
> class C
>   def foo
>     p C
>   end
> end
>
> module R1
>   refine C do
>     def foo
>       p R1
>       super
>     end
>   end
> end
>
> using R1 # 1
>
> module R2
>   using R1 # 2
>
>   refine C do
>     # using R1 # 3
>
>     def bar
>       C.new.foo
>     end
>   end
> end
>
> using R2
>
> C.new.bar
> ```
>
> Without using R1 # 3, C#foo was called in R2::C#bar.
>
> By using R1 #1 and #2, we declared that this lexical scope should use R1. However, it seems that this declaration is ignored.
>
> Is it an intentional behavior?

It's intentional.

In refine blocks of a module X, all refinements defined in X are activated, and other refinements previously activated are deactivated.