

Ruby - Bug #19079

Modules included in a DelegateClass cannot override delegate methods

10/23/2022 03:01 AM - jonathanhefner (Jonathan Hefner)

Status:	Rejected	
Priority:	Normal	
Assignee:		
Target version:		
ruby -v:	ruby 3.1.2p20	Backport: 2.7: UNKNOWN, 3.0: UNKNOWN, 3.1: UNKNOWN
Description Because DelegateClass defines delegate methods on the class itself, those delegate methods come first in the method lookup chain. This prevents included modules from overriding delegate methods: <pre>Base = Class.new do def foo "base" end end Helper = Module.new do def foo "helper" end end WithHelper = DelegateClass(Base) { include Helper } WithHelper.new(Base.new).foo # => "base"</pre> One possible solution would be to define the delegate methods in a separate module. That way, other modules could come before it in the method lookup chain.		
Related issues: Related to Ruby - Bug #19047: DelegateClass displays "method redefined" warni... Closed		

History

#1 - 10/23/2022 03:02 AM - jonathanhefner (Jonathan Hefner)

I have submitted a PR with the proposed solution: <https://github.com/ruby/delegate/pull/14>.

#2 - 12/01/2022 05:11 AM - matz (Yukihiro Matsumoto)

- Status changed from Open to Rejected

The DelegateClass defines an anonymous class and defines forwarding methods to the class. The reported (so-called) issue is a natural consequence of the above behavior. include add methods defined in a module **above** the current class, so forwarding methods have higher precedence. If you (re)define a method, it overwrites the forwarding method.

My opinion is that your assumption is wrong, so we don't need to fix. If you think we need to implement your assumption, you need to persuade us with the real world use-case.

Matz.

#3 - 12/01/2022 12:36 PM - matz (Yukihiro Matsumoto)

- Related to Bug #19047: DelegateClass displays "method redefined" warning when overriding methods added

#4 - 12/01/2022 12:38 PM - matz (Yukihiro Matsumoto)

If you want to include a module to a delegated class, try the following:

```
class WithHelper<DelegateClass(Base)
  include Helper
end
```

```
end
```

Matz.

#5 - 12/01/2022 02:57 PM - Hanmac (Hans Mackowiak)

you can also try prepend instead of include:

```
WithHelper = DelegateClass(Base) { prepend Helper }
```

#6 - 12/01/2022 05:19 PM - jonathanhefner (Jonathan Hefner)

This issue occurred for a private module in Rails: https://github.com/rails/rails/pull/46189#discussion_r991440668.

Using include in a subclass works. Using prepend also works, and is the workaround I used for the Rails module.

However, my proposed solution for this issue (<https://github.com/ruby/delegate/pull/14>) also solves [#19079](#) with a performance improvement.

I opened this issue and [#19079](#) because the current behavior seemed surprising to me. In particular, I expected the DelegateClass block to behave just like a Class.new block. I feel like that is a reasonable assumption based on [the documentation](#).

But, if my assumption is wrong, then I understand the decision.