Ruby - Feature #21028

Method for finding why an object isn't Ractor shareable

01/11/2025 11:24 PM - tenderlovemaking (Aaron Patterson)

Status:	Feedback	
Priority:	Normal	
Assignee:	tenderlovemaking (Aaron Patterson)	
Target version:		

Description

Ractor.shareable? is easy to use, but if it returns false I would like to be able to figure out what object is causing the data structure to *not* be Ractor shareable.

The context is that I'm trying to make some complex data structures in Rails deeply frozen. If they are deeply frozen they *should* be Ractor shareable, but Ractor.shareable? is returning false and it's hard for me to figure out *why*.

I would like a method that would either return all unshareable references, or a method that takes a block and unshareable references are yielded to the block.

A method like Ractor.unshareable_references? or maybe Ractor.shareable?(obj) { |not_shareable_obj| } would be very helpful for discovering why an object is not shareable.

Thanks!

History

#1 - 01/13/2025 06:46 PM - tenderlovemaking (Aaron Patterson)

- Assignee set to tenderlovemaking (Aaron Patterson)

#2 - 01/13/2025 10:47 PM - luke-gru (Luke Gruber)

I suspect it's probably a Proc object. The feature sounds good to me, and wouldn't be hard to implement I think.

#3 - 02/01/2025 04:33 PM - osyoyu (Daisuke Aritomo)

+1 for this feature. It is quite hard to find out that a rb_data_type_t in the deep is lacking the RUBY_TYPED_FROZEN_SHAREABLE flag.

#4 - 02/13/2025 09:21 AM - mame (Yusuke Endoh)

Is ObjectSpace.reachable_objects_from usable for the use case? It's a bit tedious, but I think it's more flexible not only to get an unshareable object, but also to get the path to the object in question.

```
require "objspace"

def find_path_to_unshareable_object(obj, path = [], visited = {})
  path += [obj]
  return if visited[obj]
  visited[obj] = true

  return if Ractor.shareable?(obj)

  objs = ObjectSpace.reachable_objects_from(obj)
  return path if objs.all? { Ractor.shareable?(it) }

  objs.each do |obj2|
    res = find_path_to_unshareable_object(obj2, path, visited)
    return res if res
  end
  return nil
end

pp *find_path_to_unshareable_object([1, 2, 3, "str".freeze, { :key => -> {} }])
#=> [1, 2, 3, "str", {key: #<Proc:0x00007f0a38e0ff30 t.rb:20 (lambda)>}]
#=> {key: #<Proc:0x00007f0a38e0ff30 t.rb:20 (lambda)>}
#=> #<Proc:0x00007f0a38e0ff30 t.rb:20 (lambda)>
#=> main
```

#5 - 03/13/2025 09:10 AM - mame (Yusuke Endoh)

- Status changed from Open to Feedback