Ruby - Feature #14332

Module.used_refinements to list refinement modules

01/07/2018 11:31 PM - Eregon (Benoit Daloze)

Status:	Closed
Priority:	Normal
Assignee:	shugo (Shugo Maeda)
Target version:	

Description

Module.used_modules was added in #7418.
But I think Module.used_refinements is more useful or at least complementary.

Refinements were implemented in TruffleRuby, and I found Module.used_refinements so useful that I left it there.

Instead of listing namespace modules (arguments to #using), it lists the refinement modules:

```
module Json
  refine Integer do
    def to_json
      to_s
    end
  end

  refine String do
    def to_json
      inspect
    end
  end
end

module Fact
  refine Integer do
    def fact
      self <= 1 ? 1 : self * (self-1).fact
    end
  end
end

using Json
p Module.used_modules # => [Json]
p Module.used_refinements # => [#<refinement:Integer@Json>, #<refinement:String@Json>]

using Fact
p Module.used_modules # => [Json, Fact]
p Module.used_refinements
# => [#<refinement:Integer@Fact>, #<refinement:Integer@Json>, #<refinement:String@Json>]
```

This shows which classes are refined and by which refinement namespace.
It also shows if a class has multiple active refinements in the scope.
And, last but not least, the name of the method contains "refinements".
I find used_modules hard to remember and it doesn't sound related to refinements from the name (while looking at the Module's methods).

Related issues:

Related to Ruby - Feature #12737: Module#defined_refinements	**Closed**

History

#1 - 02/19/2018 10:02 AM - shugo (Shugo Maeda)

- Status changed from Open to Assigned

Eregon (Benoit Daloze) wrote:

> Module.used_modules was added in #7418.
> But I think Module.used_refinements is more useful or at least complementary.

I'm for it, but I'd like to hear others' opinions.

#2 - 12/10/2018 06:59 AM - naruse (Yui NARUSE)

- Target version deleted (2.6)

#3 - 10/22/2021 05:18 AM - shugo (Shugo Maeda)

- Assignee changed from shugo (Shugo Maeda) to matz (Yukihiro Matsumoto)

Matz, can I add Module.used_refinements?

#4 - 10/26/2021 07:10 PM - Eregon (Benoit Daloze)

- Related to Feature #12737: Module#defined_refinements added

#5 - 10/28/2021 11:16 AM - Eregon (Benoit Daloze)

- Description updated

#6 - 11/19/2021 02:14 AM - shugo (Shugo Maeda)

Matz accepted Module.used_refinements at the developers meeting on 2021-11-18.

But we need to consider consistency of the return value with #12737, used_refinements will be introduced after #12737 is accepted (maybe in Ruby 3.2?).

#7 - 11/19/2021 08:23 PM - Eregon (Benoit Daloze)

For Module.used_refinements, I believe it has to be an Array, not a Hash.
We can see in the description:

```
p Module.used_refinements
# => [#<refinement:Integer@Fact>, #<refinement:Integer@Json>, #<refinement:String@Json>]
```

The refined classes are not unique, so we can't key by refined class.
And the order matters, so a Hash[namespace => [refined classes]] would be bad and needlessly complicated.

Since the spec of Module.used_refinements seems clear, I suggest to add it soon.

#8 - 11/19/2021 11:38 PM - shugo (Shugo Maeda)

Eregon (Benoit Daloze) wrote in #note-7:

> For Module.used_refinements, I believe it has to be an Array, not a Hash.
> We can see in the description:
>
> ```
> p Module.used_refinements
> # => [#<refinement:Integer@Fact>, #<refinement:Integer@Json>, #<refinement:String@Json>]
> ```
>
> The refined classes are not unique, so we can't key by refined class.
> And the order matters, so a Hash[namespace => [refined classes]] would be bad and needlessly complicated.

Agreed.

> Since the spec of Module.used_refinements seems clear, I suggest to add it soon.

Matz, what do you think of it?

https://github.com/shugo/ruby/compare/used_refinements

#9 - 12/09/2021 07:15 AM - matz (Yukihiro Matsumoto)

It's too late for 3.1. But after the release, I basically honor @shugo's decision here.

Matz.

#10 - 12/10/2021 01:48 AM - shugo (Shugo Maeda)

- Assignee changed from matz (Yukihiro Matsumoto) to shugo (Shugo Maeda)

#11 - 01/05/2022 08:07 AM - shugo (Shugo Maeda)

- Status changed from Assigned to Closed

Applied in git|21ee5341f8fc4ca513295dff2148f7c203c908a7.