I'd like to propose a method that returns an array of all refinements visible from the current scope. I have documented the ordering of the return array as undefined to allow a bit of flexibility in how it is implemented across the various Ruby implementations.

One of the criticisms of refinements is that its use makes code harder to understand - especially when non-lexical refinements are being used. This method slightly improves the situation by making debugging of code using refinements easier.

I've named the method 'used_refinements' for now, but I'm open to changing that to something else.

I've already done the initial implementation which is attached as a patch. Feedback would be most appreciated.
• eval.c (rb_mod_s_used_modules): rename Module.used_refinements to Module.used_modules. [Feature #7418] [ruby-core:49805]

Revision 56109 - 09/08/2016 07:49 AM - shugo (Shugo Maeda)

• eval.c (rb_mod_s_used_modules): rename Module.used_refinements to Module.used_modules. [Feature #7418] [ruby-core:49805]

Revision 56109 - 09/08/2016 07:49 AM - shugo (Shugo Maeda)

• eval.c (rb_mod_s_used_modules): rename Module.used_refinements to Module.used_modules. [Feature #7418] [ruby-core:49805]

History

#1 - 11/21/2012 10:53 PM - judofyr (Magnus Holm)
Maybe it should be called Module.refinements (a la Module.nesting).

// Magnus Holm

On Wed, Nov 21, 2012 at 12:16 PM, charliesome (Charlie Somerville)
charlie@charliesomerville.com wrote:

Issue #7418 has been reported by charliesome (Charlie Somerville).

Feature #7418: Kernel#used_refinements
https://bugs.ruby-lang.org/issues/7418

Author: charliesome (Charlie Somerville)
Status: Open
Priority: Normal
Assignee:
Category:
Target version:

I'd like to propose a method that returns an array of all refinements visible from the current scope. I have documented the ordering of the return array as undefined to allow a bit of flexibility in how it is implemented across the various Ruby implementations.

One of the criticisms of refinements is that its use makes code harder to understand - especially when non-lexical refinements are being used. This method slightly improves the situation by making debugging of code using refinements easier.

I've named the method ‘used_refinements’ for now, but I'm open to changing that to something else.

I've already done the initial implementation which is attached as a patch. Feedback would be most appreciated.

--
http://bugs.ruby-lang.org/

#2 - 11/22/2012 01:23 PM - shugo (Shugo Maeda)
- Status changed from Open to Assigned
- Assignee set to shugo (Shugo Maeda)

judofyr (Magnus Holm) wrote:

Maybe it should be called Module.refinements (a la Module.nesting).

Module.refinements conflicts with Module#refinements, which returns refinements defined in the receiver.

#3 - 11/22/2012 03:51 PM - chendo (Jack Chen)
+1 since I gave Charlie the idea :)

#4 - 11/24/2012 11:51 AM - mame (Yusuke Endoh)
- Target version set to 2.6

Shugo, is this feature mandatory for the 2.0.0 refinements?
I'm setting to next minor tentatively.
I'm against adding a new reflection API for refinements into Ruby 2.1. What do you think of it, Matz?

Module.used_refinements accepted.

Module.used_modules might be a better name, because a module can have more than one refinement:

```ruby
module M
  refine A
  end

  refine B
  end
end

using M
p Module.used_modules #=> [M]
```

Matz said that he thinks so too at https://twitter.com/yukihiro_matz/status/773787246990196736, so I'll change the name.

Module.used_modules might be a better name, because a module can have more than one refinement:

Matz said that he thinks so too at https://twitter.com/yukihiro_matz/status/773787246990196736, so I'll change the name.

Module.used_modules might be a better name, because a module can have more than one refinement:

Matz said that he thinks so too at https://twitter.com/yukihiro_matz/status/773787246990196736, so I'll change the name.

Module.used_modules might be a better name, because a module can have more than one refinement:

Matz said that he thinks so too at https://twitter.com/yukihiro_matz/status/773787246990196736, so I'll change the name.