Ruby master - Bug #3169

RDoc crossref confused by instance and class methods having same name

04/19/2010 09:39 AM - marcandre (Marc-Andre Lafortune)

Status:	Closed		
Priority:	Normal		
Assignee:	drbrain (Eric Hodel)		
Target version:	1.9.2		
ruby -v:	ruby 1.9.2dev (2010-04-19 trunk 27394) [x86_64-darwin10.3.0]	**Backport:**	

Description

=begin
The documentation for the two methods below will both have a reference to X.foo (which appeared first). The "See X#foo" should reference to the instance method instead.

class X
# The class method. See X#foo
def self.foo
end

```
# The instance method. See X.foo
def foo
end
```

end
=end

History

#1 - 04/19/2010 12:36 PM - drbrain (Eric Hodel)

=begin
Fixed in rdoc trunk.

RDoc now maps :: to class methods and # and . to instance methods when cross-referencing.
=end

#2 - 04/19/2010 01:28 PM - murphy (Kornelius Kalnbach)

=begin
On 19.04.10 05:36, Eric Hodel wrote:

> RDoc now maps :: to class methods and # and . to instance methods
> when cross-referencing.
> Mapping Foo.bar to Foo#bar is a strange decision, in my view. Foo.bar
> should be equal to Foo::bar.

[murphy]

=end

#3 - 04/19/2010 02:08 PM - drbrain (Eric Hodel)

- Status changed from Assigned to Closed

=begin

=end

#4 - 04/19/2010 02:57 PM - drbrain (Eric Hodel)

=begin
On Apr 18, 2010, at 21:28, Kornelius Kalnbach wrote:

> On 19.04.10 05:36, Eric Hodel wrote:

> > RDoc now maps :: to class methods and # and . to instance methods
> > when cross-referencing.
> > Mapping Foo.bar to Foo#bar is a strange decision, in my view. Foo.bar
> > should be equal to Foo::bar.

Mapping Foo.bar to Foo#bar for cross references is backwards compatible.

Previous versions of ri displayed "Foo::bar" when bar was a class method (currently RDoc display what you typed in, a future version will display :: for class methods again). If you don't know whether the method you're looking for is an instance method or a class method '.' allows you to let ri do the work of figuring it out.

I've seen no de-facto mapping in the community of Foo.bar to either Foo::bar or Foo#bar.

In a future version RDoc cross-references will map Foo.bar to Foo::bar if there is no Foo#bar like ri. This will make RDoc's convention of using '.' to mean either instance method or class method universal.

=end

#5 - 04/20/2010 12:54 PM - tomonacci (Tomo Kazahaya)

=begin
On Mon, Apr 19, 2010 at 1:56 AM, Eric Hodel drbrain@segment7.net wrote:

> I've seen no de-facto mapping in the community of Foo.bar to either
> Foo::bar or Foo#bar.

In Japanese Ruby community, it's likely that Foo.bar refers to Foo::bar, the
class method.
For instance, in Japanese Ruby Reference Manual (
http://doc.okkez.net/192/view/ -- it's in Japanese, sorry),
Foo.bar is the primary way to describe Foo::bar.

In addition, Foo.#bar is used to describe a module function bar of module
Foo.

On Mon, Apr 19, 2010 at 1:56 AM, Eric Hodel <drbrain@segment7.net> wrote:
I've seen no de-facto mapping in the community of Foo.bar to either Foo::bar or Foo#bar.In Japanese Ruby community, it's likely that Foo.bar refers to Foo::bar, the class method.For instance, in Japanese Ruby Reference Manual (http://doc.okkez.net/192/view/ -- it's in Japanese, sorry),
Foo.bar is the primary way to describe Foo::bar.In addition, Foo.#bar is used to describe a module function bar of module Foo.

=end

#6 - 04/20/2010 04:05 PM - drbrain (Eric Hodel)

=begin
On Apr 19, 2010, at 20:53, Tomo Kazahaya wrote:

> On Mon, Apr 19, 2010 at 1:56 AM, Eric Hodel drbrain@segment7.net wrote:
>
> > I've seen no de-facto mapping in the community of Foo.bar to either Foo::bar or Foo#bar.
>
> In Japanese Ruby community, it's likely that Foo.bar refers to Foo::bar, the class method.
> For instance, in Japanese Ruby Reference Manual (http://doc.okkez.net/192/view/ -- it's in Japanese, sorry),
> Foo.bar is the primary way to describe Foo::bar.
>
> In addition, Foo.#bar is used to describe a module function bar of module Foo.

I will try to make RDoc's crossref support both class methods and instance methods sooner rather than later, hopefully for inclusion in 1.9.2. (I need to focus on RubyGems.)

I have not heard of Foo.#bar. I can make RDoc support this as well. Where can I find code samples that use this? Japanese comments are ok.

=end