Ruby - Bug #11274

Equality inconsistency between Method and UnboundMethod

06/17/2015 08:01 AM - ko1 (Koichi Sasada)

Status:	Closed		
Priority:	Normal		
Assignee:	matz (Yukihiro Matsumoto)		
Target version:			
ruby -v:	2.3dev	**Backport:**	2.0.0: UNKNOWN, 2.1: UNKNOWN, 2.2: UNKNOWN

Description

Method と UnboundMethod の equality が一貫していません。

```
module M1
  def foo; end
end

module M2
  include M1
  alias bar foo
end

class C
  include M1
  include M2
end

c = C.new
c_m1 = c.method(:foo)
c_m2 = c.method(:bar)
p [c_m1, c_m2, c_m1 == c_m2, c_m2 == c_m1]
```

結果

```
[#<Method: C(M1)#foo>, #<Method: C(M1)#bar(foo)>, true, true]
```

こんな感じで、同じと判断されます。

しかし、UnboundMethod#== を見てみると、

```
module M1
  def foo; end
end

module M2
  include M1
  alias bar foo
end

ubm1 = M1.instance_method(:foo)
ubm2 = M2.instance_method(:bar)

p [ubm1, ubm2, ubm1==ubm2, ubm2==ubm1]
```

結果

```
[#<UnboundMethod: M1#foo>, #<UnboundMethod: M2(M1)#bar(foo)>, false, false]
```

等しくない、と言われます。

ここで、異なる必要は無いと思うので、どちらかに合わせるといいと思うのですが、どうでしょうか。

ちなみに、Method#== の説明では、

```
 * Two method objects are equal if they are bound to the same
 * object and refer to the same method definition and their owners are the
 * same class or module.
```

とあり「オーナーが一緒なら」とありますが、

```
module M1
  def foo; end
end

module M2
  include M1
  alias bar foo
end

class C
  include M1
  include M2
end

c = C.new
c_m1 = c.method(:foo)
c_m2 = c.method(:bar)

p c_m1.owner, c_m2.owner
```

結果

```
M1
M2
```

と、オーナーが異なります。なので、Method#== の挙動か、ドキュメントが間違いなんじゃないかと思います。

多分、あまり影響が無いので、誰も気にしていないのじゃ無いかとは思うんですが、誰か一家言ある人は居ますかね。
とりあえず、松本さんに振っておきますが、とくにご意見がないようでしたら、実装が簡単な方（owner が異なれば否定）としようと思います。

alias されたメソッド同士は同値であるか、という問題です。

History

#1 - 06/17/2015 03:12 PM - matz (Yukihiro Matsumoto)

一致する方に揃えると良いと思います。
また、ドキュメントはaliasを考慮していないと思うので、「ownerの一致または同じ実体を指すalias」とでも記述するのが良いのではないでしょうか。

Matz.

#2 - 10/21/2019 05:42 PM - jeremyevans0 (Jeremy Evans)

- Status changed from Open to Closed

This appears to be fixed starting in Ruby 2.3 (the first example returns false instead of true for equality as the owners of the methods are different), probably by 5e8a147480f87f19a8b96ad3fb33a25fb4bb19b9.