# Adding Numeric#divisor? (Have working implementation)

**07/21/2010 03:28 PM - duckinator (Marie Markwell)**

<table>
<thead>
<tr>
<th>Status</th>
<th>Rejected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee</td>
<td>matz (Yukihiro Matsumoto)</td>
</tr>
<tr>
<td>Target version</td>
<td>2.6</td>
</tr>
</tbody>
</table>

**Description**

=end

On the 'duckinator' branch of my fork of ruby on github, I have added Numeric#divisor?:

num.divisor?(other)

is the same as:

(num % other == 0)

Example usage:

4.divisor?(2) #=> true
1.divisor?(2) #=> false
2.4.divisor?(1.2) #=> true
2.4.divisor?(1.3) #=> false
126.divisor?(9) #=> true

I think this would be a very nice feature to add, and would make code using the old (num % other == 0) method much cleaner and easier to understand.

There is a unified diff of it: [http://gist.github.com/raw/484144/07b1a6e696cd9301e658ccbc8f90dfcd4d4ef3f1/Numeric_divisor.patch](http://gist.github.com/raw/484144/07b1a6e696cd9301e658ccbc8f90dfcd4d4ef3f1/Numeric_divisor.patch)

The original commits can be seen here:

[http://github.com/RockerMONO/ruby/commit/f7959f964cb0bf38418904cc5643db6b689d29c](http://github.com/RockerMONO/ruby/commit/f7959f964cb0bf38418904cc5643db6b689d29c) -- First attempt, only worked with Integers

[http://github.com/RockerMONO/ruby/commit/12376a6bb1c3ffbd4b470850dd758e2dcd783dda](http://github.com/RockerMONO/ruby/commit/12376a6bb1c3ffbd4b470850dd758e2dcd783dda) -- Second attempt, should work with anything derived from Numeric that can be used as (num % other == 0)

=end

**History**

**#1 - 07/21/2010 03:35 PM - shyouhei (Shyouhei Urabe)**

=end

I think this would be a very nice feature to add, and would make code using the old (num % other == 0) method much cleaner and easier to understand.

+1, I have written tons of such code.

=end

**#2 - 07/21/2010 04:22 PM - nobu (Nobuyoshi Nakada)**

=end

Hi,

At Wed, 21 Jul 2010 15:28:35 +0900,

Nick Markwell wrote in [ruby-core:31385]:

Example usage:

4.divisor?(2) #=> true
1.divisor?(2) #=> false
2.4.divisor?(1.2) #=> true
2.4.divisor?(1.3) #=> false
126.divisor?(9) #=> true

I suspect if Float#divisor? makes a sense, and the name doesn't feel appropriate very much. The first example seems four is a divisor of two? What about4.multiple?(2)?

03/20/2022 1/5
I agree, Numeric#multiple? seems more appropriate.
Different patch providing Numeric#multiple? with the same usage:
http://gist.github.com/raw/484193/65ffa0cb48ca4b838900dc105b246ced5bc0ac80/Numeric_multiple.patch
Or Numeric#divisible?, I'm not sure which is better. Maybe define both?

And, it's arguable about inexact numbers, I guess.

--
Nobu Nakada
=end

#6 - 07/21/2010 11:35 PM - hasari (Hiro Asari)
=begin
On Jul 21, 2010, at 3:55 AM, Nobuyoshi Nakada wrote:

Hi,

At Wed, 21 Jul 2010 16:35:49 +0900,
Nick Markwell wrote in [ruby-core:31391]:

 I agree, Numeric#multiple? seems more appropriate.

Or Numeric#divisible?, I'm not sure which is better. Maybe define both?

And, it's arguable about inexact numbers, I guess.

--
Nobu Nakada

"divisible" and "divisor" should not mean the same thing, and I vote for Numeric#divisible? (or more verbosely, Numeric#divisible_by?) for the present concept under discussion.

When I see "10.divisor? 5", it reads to me "Is 10 a divisor of 5?", just like "h.kind_of? Hash" reads "Is h a kind of Hash?".

On the other hand, "10.divisible? 5" reads "Is 10 divisible by 5?".
=.end

#7 - 07/22/2010 05:54 AM - duckinator (Marie Markwell)
=begin
Toss out some more ideas/opinions, I'll make a patch for the one that the most people agree with :)
=.end

#8 - 06/26/2011 01:39 PM - akr (Akira Tanaka)
- Project changed from Ruby to Ruby master

#9 - 03/25/2012 02:13 PM - mame (Yusuke Endoh)
- Description updated
- Status changed from Open to Assigned
- Assignee set to mrkn (Kenta Murata)

#10 - 07/13/2012 12:31 PM - duckinator (Marie Markwell)
=begin
I ran across this again, and decided to just implement it as Numeric#divisible_by?, since that makes the most sense to me.

Branch on github: (URL https://github.com/duckinator/ruby/compare/feature/numeric_divisible_by)
Patch: (URL https://github.com/duckinator/ruby/compare/feature/numeric_divisible_by.patch)
Diff: (URL https://github.com/duckinator/ruby/compare/feature/numeric_divisible_by.diff)
=end

#11 - 07/13/2012 02:11 PM - marcandre (Marc-Andre Lafortune)

Hi,
Agreed. divisible_by makes sense only for Integers. Float, BigDecimal, Complex... and Rational which can be exact, for them it does not make sense. I am not convinced about the need for this. There is already odd? or even?, and for other divisors I feel usage is much less frequent.

In those cases 42 % 5 == 0 is shorter to write and faster than 42.divisible_by?(5). Unless we talk about big integers in which case there might be an optimization possible, but I feel that falls outside of the core of Ruby.

#12 - 07/13/2012 02:50 PM - duckinator (Marie Markwell)
Regarding Float, BigDecimal, Complex, and Rational: Agreed.

Regarding 42 % 5 == 0, I somewhat agree: it can be accomplished now, and it is not done often (excluding x % 2 == 0), so it may not be a good fit for core Ruby. When splitting it out so it does not cover things it does not apply to, it'd likely become more complex than it is worth dealing with.

#13 - 07/13/2012 10:41 PM - naruse (Yui NARUSE)
We talked about this before, and find the name Numeric#factor?(n).

#14 - 07/13/2012 11:51 PM - marcandre (Marc-Andre Lafortune)
I still curious as to actual use cases. Looking at Rails' code, there is one instance of this, and followed immediately by the more general foo % bar == baz (when implementing nth_child).

Also, I see that ActiveSupport already has 42.multiple_of?(2), and that's a good name too. It's not used in Rails itself, btw.

naruse (Yui NARUSE) wrote:

We talked about this before, and find the name Numeric#factor?(n).

2 is a factor of 4, so which is true: 4.factor?(2) or 2.factor?(4)? There is no ambiguity with divisible? or multiple_of?

#15 - 07/13/2012 11:52 PM - marcandre (Marc-Andre Lafortune)
- Category set to core
- Assignee changed from mrkn (Kenta Murata) to matz (Yukihiro Matsumoto)

#16 - 11/20/2012 11:12 PM - mame (Yusuke Endoh)
- Target version set to 2.6

#17 - 11/27/2017 12:24 PM - mame (Yusuke Endoh)
marcandre (Marc-Andre Lafortune) wrote:

I still curious as to actual use cases. Looking at Rails' code, there is one instance of this, and followed immediately by the more general foo % bar == baz (when implementing nth_child).

With regard to this feature, there is no room for doubt about use cases, I think. I don't know Rails, but this method would be frequently useful when enjoying mathematical programming. I can find many actual use cases in my programs for Project Euler:

$ egrep "\% \w+ == 0" ~/work/peuler/*.rb | wc -l
86

Of course, this feature is NOT indispensable, though.

#18 - 11/27/2017 12:31 PM - mame (Yusuke Endoh)
Summary:

- Name candidates are:
  - divisor? (the original proposal)
  - divisible?
  - divisible_by? (OP's final choice)
  - multiple?
  - multiple_of? (ActiveSupport's choice)
  - factor?
- There is no objection to Integer#divisor?. But Float#divisor? is arguable.
Use-case? Is it more useful than n % m == 0?

Matz.