Ruby - Feature #7580

Range translation

12/17/2012 02:43 PM - Anonymous

Status:	Assigned
Priority:	Normal
Assignee:	matz (Yukihiro Matsumoto)
Target version:	

Description

=begin
I would like to propose the (({#+})) and (({#-})) methods on (({Range})).

These would be useful for translating ranges - for example, given a range where the endpoints are 1-indexed, the range could be translated by 1 in the negative direction to use in (({Array#[]})).

Instead of doing a syntactically-bulky manual translation like so:

ary[(range.begin - 1)..(range.end - 1)]

(({Range#-})) could be used instead:

ary[range - 1]

The translation methods would not handle certain endpoint types specially, they would just pass the call on.

Here's an example implementation in Ruby:

class Range
def +(other)
Range.new(self.begin + other, self.end + other, exclude_end?)
end

```
def -(other)
  Range.new(self.begin - other, self.end - other, exclude_end?)
end
```

end

=end

History

#1 - 12/17/2012 02:57 PM - naruse (Yui NARUSE)

I think such arithmetic is not addition/subtraction, but shift.

#2 - 12/17/2012 03:02 PM - Anonymous

=begin
Do you propose that (({Range#<<})) would use (({#-})) and (({Range#>>})) would use (({#+})), or would it be a different method call internally?

I am happy with both alternatives, I just want nice convenience methods for this operation.
=end

#3 - 12/17/2012 03:03 PM - dummey (Ricky Ng)

I would normally agree that 'shift' would be the proper term except that it's used in Array already which could cause a bit of confusion.

#4 - 12/17/2012 04:14 PM - naruse (Yui NARUSE)

- Status changed from Open to Assigned

- Assignee set to matz (Yukihiro Matsumoto)

charliesome (Charlie Somerville) wrote:

> Do you propose that (({Range#<<})) would use (({#-})) and (({Range#>>})) would use (({#+})), or would it be a different method call internally?

Don't use +/- and use <</>> or Range#shift().

> I am happy with both alternatives, I just want nice convenience methods for this operation.

In my experience, such alternative name considered harmful
because if you want to add another method as Arange#+ in the future, those aliases prevent it.

#5 - 01/25/2013 11:15 PM - Anonymous

- Target version set to 2.6

#6 - 12/25/2017 06:15 PM - naruse (Yui NARUSE)

- Target version deleted (2.6)

#7 - 06/11/2018 09:49 AM - DanielRHeath (Daniel Heath)

Duplicated at https://bugs.ruby-lang.org/issues/14777

Agree that '+' seems an idiomatic name (I'm using ranges to represent coordinate ranges for bounding boxes).

It's somewhat awkward given that a Range can be eg ('a'..'z') - should ('a'..'z') + 'b' return ('ab'..'zb')?

#8 - 06/11/2018 09:51 AM - DanielRHeath (Daniel Heath)

On the other hand, + could easily be construed to construct non-contiguous ranges (eg ((1..2) + (4..5)).to_a == [1,2,4,5].