

Ruby - Bug #4245

Array index with Range inconsistency

01/07/2011 01:14 AM - kristianpd (Kristian PD)

Status:	Rejected	
Priority:	Normal	
Assignee:		
Target version:		
ruby -v:	1.8.7	
Description		Backport:
=begin I've come across what I believe to be a possible inconsistency in the implementation of the Range array index method. The following example describes my case: array = [1,2,3,4,5] tests a) array[0..5] // [1,2,3,4,5] b) array[4..-1] // [5] c) array[5..-1] // [] d) array[6..-1] // nil additional tests e) array[5..10] // [] f) array[6..10] // nil I would have expected cases c and d to both return nil according to the docs: "Returns nil if the index (or starting index) are out of range." Out of curiosity I also added tests e and f to make sure it was not related to the end of the range being -1. Without having had a chance to check the code, my instinct here is that a comparison is possibly being done on the .count/.length of the array when building the result and comparing that to the first part of the range sequence. Am I missing another reason for this? Kristian =end		
Related issues:		
Has duplicate Ruby - Feature #4541: Inconsistent Array.slice()		Rejected

History

#1 - 01/07/2011 04:28 AM - marcandre (Marc-Andre Lafortune)

- Status changed from Open to Rejected

=begin

This is per spec.

Think of it this way:

[5..-1]: give me all elements after the 5th one and up to the last one. There are none, so []

[6..-1]: give me all elements after the 6th one and up to the last one. Say what? There is no 6th one, so returns nil as the given request is out of bounds.

=end

#2 - 01/07/2011 07:42 PM - sdsykes (Stephen Sykes)

=begin

Yes, it's a bit odd, but it is clearly noted in the docs that this special case exists.

Marc-Andre: - For me at least I'm not sure that thinking about it as 'the element after the nth one' is totally helpful - eg:

```
a = []  
=> []  
a[1..1]  
=> nil  
a[0..0]  
=> []
```

But whichever way you think about it, this is how it works.

From memory (it has been raised before) I think the reason was that it can make handling end conditions easier if an empty array is returned rather than nil if you hit the first non existing element after the end of the array.

=end