Ruby trunk - Feature #6367

#same? for Enumerable

04/26/2012 10:35 PM - prijutme4ty (Ilya Vorontsov)

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

Description

I realised that I frequently test if all objects in enumerable have the same feature. For example if all words have the same length (but not defined before). So I found particulary useful method Enumerable#same_by? that test this behaviour. I think it can be simply rewritten in C and included to Enumerable core methods.
Simple ruby implementation can be written just in a pair of lines (tests included):

```
module Enumerable
def same?(&block)
return true if empty?
if block_given?
first_result = yield(first)
all?{|el| first_result == yield(el)}
else
first_result = first
all?{|el| first_result == el}
end
end
end

require 'test/unit'
class TestEnumerableSame < Test::Unit::TestCase
def test_same
assert_equal(true, [1,3,9,7].same?(&:even?))
assert_equal(true, [4,8,2,2].same?(&:even?))
assert_equal(false, [1,8,3,2].same?(&:even?))

assert_equal(true, %w{cat dog rat}.same?(&:length))
assert_equal(false, %w{cat dog rabbit}.same?(&:length))

assert_equal(true, %w{cat cat cat}.same?)
assert_equal(false, %w{cat dog rat}.same?)

assert_equal(true, [].same?(&:length))
assert_equal(true, [].same?)

end
end
```

History

#1 - 04/27/2012 03:21 AM - mame (Yusuke Endoh)

- Status changed from Open to Assigned

- Assignee set to matz (Yukihiro Matsumoto)

Personally I don't think it is a good name.
It looks a kind of comparison operator.

--
Yusuke Endoh mame@tsg.ne.jp

#2 - 04/27/2012 04:41 AM - prijutme4ty (Ilya Vorontsov)

I think any name you choose can be used.

#3 - 04/27/2012 02:18 PM - matz (Yukihiro Matsumoto)

- Status changed from Assigned to Feedback

I accept the idea of having the method that ensures all elements are same (under some criteria).
But the same? is not good name for it. I place this "feedback" to get the "right" name.

Matz.

#4 - 04/27/2012 05:06 PM - alexeymuranov (Alexey Muranov)

homologous? :)

#5 - 04/28/2012 03:12 PM - edtsech (Edward Tsech)

I don't know, but I was a little bit confused by that:
assert_equal(true, [1,3,9,7].same?(&:even?))
assert_equal(true, [4,8,2,2].same?(&:even?))

or I can write smth like that:
assert_equal(true, [1.0,2.0,3.0,4.0].same?(&:integer?))

Maybe it's just unusable for "primitive" types,
because as i understand this method useful for test if all objects in enumerable have the same "property",
and "primitive" types actually don't have them.

But I like this idea:
assert_equal(true, %w{cat dog rat}.same?(&:length))
assert_equal(false, %w{cat dog rabbit}.same?(&:length))

#6 - 04/28/2012 08:46 PM - trans (Thomas Sawyer)

This reminds me of #sort and #sort_by, and I think both forms would be needed here too --where the later makes it possible to use a comparison besides #==.

On the other hand it reminds me of #all? as well, which makes me wonder about #any? too.

That leads me to think:

#all_equal?{ |a| ... }
#all_equal_by?{ |a,b| ... }
#any_equal?{ |a| ... }
#any_equal_by?{ |a,b| ... }

#7 - 04/29/2012 01:45 AM - prijutme4ty (Ilya Vorontsov)

edtsech (Edward Tsech) wrote:

> I don't know, but I was a little bit confused by that:
> assert_equal(true, [1,3,9,7].same?(&:even?))
> assert_equal(true, [4,8,2,2].same?(&:even?))
>
> or I can write smth like that:
> assert_equal(true, [1.0,2.0,3.0,4.0].same?(&:integer?))

I think renaming of method can make it less confusing. May be Enumerable#coincident?/coincident_by? or Enumerable#identical_by?

#8 - 04/29/2012 01:49 AM - prijutme4ty (Ilya Vorontsov)

trans (Thomas Sawyer) wrote:

> This reminds me of #sort and #sort_by, and I think both forms would be needed here too --where the later makes it possible to use a comparison besides #==.
>
> On the other hand it reminds me of #all? as well, which makes me wonder about #any? too.
>
> That leads me to think:
>
> #all_equal?{ |a| ... }
> #all_equal_by?{ |a,b| ... }
> #any_equal?{ |a| ... }
> #any_equal_by?{ |a,b| ... }

As for me it's contrintuitive to have a pair of argument. In fact we need the only argument... or maybe I didn't understand how to use this function (and how to write the method)

#9 - 04/29/2012 03:39 AM - trans (Thomas Sawyer)

=begin
My methods names for the "_by" methods are not very good. This should clarify:

```
[1,1,1].all_equal?          #=> true
[1,1,2].all_equal?          #=> false
[2,4,6].all_equal?{|x| x*0}  #=> true

[1,1,2].any_equal?          #=> true
[1,2,3].any_equal?          #=> false
[2,3,4].any_equal?(&:even?)  #=> true

[2,4,6]#all_by?{ |a,b| a % 2 == b % 2 }  #=> true
[2,3,4]#all_by?{ |a,b| a % 2 == b % 2 }  #=> false

[2,3,4]#any_by?{ |a,b| a % 2 == b % 2 }  #=> true
[1,2,3]#any_by?{ |a,b| a % 3 == b % 3 }  #=> false
```

=end

#10 - 04/30/2012 11:35 PM - pabloh (Pablo Herrero)

trans (Thomas Sawyer) wrote:

> My methods names for the "_by" methods are not very good. This should clarify:
>
> ```
> [1,1,1].all_equal?          #=> true
> [1,1,2].all_equal?          #=> false
> [2,4,6].all_equal?{|x| x*0}  #=> true
>
> [1,1,2].any_equal?          #=> true
> [1,2,3].any_equal?          #=> false
> ```

Is a bit misleading about if it's using #equal? or #== for the comparison.

#11 - 05/02/2012 07:46 PM - andhapp (Anuj Dutta)

homogeneous? :)

#12 - 05/10/2012 03:36 AM - tho119cl (Thomas Green)

I like 'same_by?'
['ten', 'six', 'one', 'two'].same_by? &:length ===> true

IMO It's better to use a simple word, such as 'same', over 'identical' or 'coincident'

#13 - 06/01/2012 03:42 PM - prijutme4ty (Ilya Vorontsov)

same_by? is a good alternative. I've even replaced #same? with #same_by? in my projects.

#14 - 11/20/2012 10:56 PM - mame (Yusuke Endoh)

- Target version set to 2.6

#15 - 04/26/2013 02:08 PM - prijutme4ty (Ilya Vorontsov)

Possible names: #same_by?, #all_equal_by?, #equal_by?, #identical_by?, #uniform_by? I think same_by? looks the most natural.
set_of_words.same_by?(&:length)
numbers.same_by{|number| number % 3 }
card_in_hand.same_by?(&:color)

#16 - 12/25/2017 06:15 PM - naruse (Yui NARUSE)

- Target version deleted (2.6)