Ruby - Feature #13676

to_s method is not overriden for Set

06/23/2017 01:09 PM - razor (Marat Chardymov)

Status:	Closed	
Priority:	Normal	
Assignee:	knu (Akinori MUSHA)	
Target version:		

Description

When I call

```
s1 = Set.new
s1<<'tic'<<'tac'
s1.to_s
```

I'd expect ['tic', 'tac'] values being printed, not "#Set:0x0055f331076348"

Associated revisions

Revision d893c123f6b021254b21c920e182d3c64967f5d5 - 07/14/2017 08:46 AM - Akinori MUSHA

Alias Set#to_s to #inspect [ruby-core:81753] [Feature #13676]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@59332 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision d893c123 - 07/14/2017 08:46 AM - Akinori MUSHA

Alias Set#to_s to #inspect [ruby-core:81753] [Feature #13676]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@59332 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 06/23/2017 01:17 PM - nobu (Nobuyoshi Nakada)

- Description updated

- Status changed from Open to Feedback

What do you want to do?
If you want a list of the elements, use to_a.
If you want to see what elements it has, use inspect.

#2 - 06/23/2017 01:47 PM - razor (Marat Chardymov)

nobu (Nobuyoshi Nakada) wrote:

> What do you want to do?
> If you want a list of the elements, use to_a.
> If you want to see what elements it has, use inspect.

I'm printing several sets in my erb. It would be nice have such **to_s** behaviour by default, eliminating the need to call **to_a** on each of them before print

#3 - 06/23/2017 01:57 PM - zverok (Victor Shepelev)

> What do you want to do?
> If you want a list of the elements, use to_a.
> If you want to see what elements it has, use inspect.

Isn't it just a decent behavior for any value object to have a readable to_s representation?.. Say, for cases like puts-debugging (puts "Comparing #{set1} to #{set2}...") or developer-friendly error messages (raise "Input was expected to have 4 elements, #{set} received").

#4 - 06/23/2017 03:55 PM - shevegen (Robert A. Heiler)

I have no pro or con opinion. I did however had want to compare Set to Array and the two behave differently.

```
require 'pp'
require 'set'

s1 = Set.new
s1<<'tic'<<'tac'
puts s1.to_s
pp s1

array = Array.new
array << 'tic' << 'tac'
puts array.to_s
pp array

# Output:
#
# <Set:0x810460c4>
# <Set: {"tic", "tac"}>
# ["tic", "tac"]
# ["tic", "tac"]
```

I have no idea why Set behaves that way, perhaps there is a clear reason.

I can however had understand razor too - without knowing the context or really having a lot of experience with Set mayself, to me the behaviour of Array seems "more useful" by default. But again, I have no real idea about this so neither can I say good or bad if it would be changed - I really don't know. I only use Arrays, barely ever Set myself. :)

#5 - 07/14/2017 08:19 AM - knu (Akinori MUSHA)

- Status changed from Feedback to Assigned

- Assignee set to knu (Akinori MUSHA)

Makes sense. I'll add to_s as an alias to inspect.

#6 - 07/14/2017 08:46 AM - knu (Akinori MUSHA)

- Status changed from Assigned to Closed

Applied in changeset trunk|r59332.

Alias Set#to_s to #inspect [ruby-core:81753] [Feature #13676]