Ruby - Feature #12210

Add IdentitySet class that compares members by identity

03/23/2016 07:26 PM - tjwp (Tim Perkins)

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

Description

This subclass of Set handles a use case that we ran into where we needed to track instances of objects that might compare as equal.

I was surprised that there was not a core way to do this. IdentitySet allows you to do the following (trivial example using strings):

```
a_str = "a"
s = IdentitySet.new([a_str, a_str, "b", "b"])

p s # => #<IdentitySet: {"a", "b", "b"}>
```

Associated revisions

Revision 76977611dd68e384fdce8c546efda5e1931e67a6 - 11/05/2016 09:23 AM - Akinori MUSHA

Add Set#compare_by_identity and Set#compare_by_identity?

- lib/set.rb (Set#compare_by_identity, Set#compare_by_identity?):
  New methods. [Feature #12210]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@56589 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 76977611 - 11/05/2016 09:23 AM - Akinori MUSHA

Add Set#compare_by_identity and Set#compare_by_identity?

- lib/set.rb (Set#compare_by_identity, Set#compare_by_identity?):
  New methods. [Feature #12210]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@56589 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 03/24/2016 01:40 AM - duerst (Martin Dürst)

I think that rather than creating a new class, adding an option on Set.new (e.g. Set.new([a_str, a_str, "b", "b"], compare_by: :identity)) is more Ruby-like.

#2 - 03/24/2016 01:47 AM - shyouhei (Shyouhei Urabe)

If you want a new class, why not start as a separate gem for it?

#3 - 03/24/2016 12:02 PM - tjwp (Tim Perkins)

I submitted this as a patch because the std lib already contains one subclass of Set (SortedSet) and a commented-out implementation of another (RestrictedSet).

I don't expect that this class will change beyond the initial implementation. Following the precedent of SortedSet, I think that including IdentitySet in the std lib improves discoverability over providing it via a gem, which seems like a more heavyweight solution for a simple, useful variation on a Set.

I was surprised that there wasn't already a standard way to do this.

#4 - 03/28/2016 03:34 AM - shyouhei (Shyouhei Urabe)

- Status changed from Open to Assigned

- Assignee set to knu (Akinori MUSHA)

#5 - 11/05/2016 09:19 AM - knu (Akinori MUSHA)

- Tracker changed from Misc to Feature

I'd rather add Set#compare_by_identity.

#6 - 11/05/2016 09:23 AM - knu (Akinori MUSHA)

- Status changed from Assigned to Closed

Applied in changeset r56589.

Add Set#compare_by_identity and Set#compare_by_identity?

- lib/set.rb (Set#compare_by_identity, Set#compare_by_identity?):
  New methods. [Feature #12210]

#7 - 11/05/2016 09:34 AM - knu (Akinori MUSHA)

Note that SortedSet#compare_by_identity is unimplemented when SortedSet uses rbtree.

rbtree.readjust { |a, b| a.object_id <=> b.object_id } comes pretty close, but rbtree treats string keys specially by storing a copy of each string key given, so it does not work as expected for string keys. There needs some way provided on the rbtree side.

Files

identity_set.diff	1.18 KB	03/23/2016	tjwp (Tim Perkins)