Ruby master - Feature #3963
Map class in standard library
10/19/2010 03:08 AM - trans (Thomas Sawyer)

Status: Rejected
Priority: Normal
Assignee: matz (Yukihiro Matsumoto)
Target version: 2.0.0

Description
=begin
Could we get a class like Ara Howard's Map (http://github.com/ahoward/map) included in Ruby's standard library?

Map is a class that would get a lot of usage. We already see similar classes in other code like Rail's with it's HashWithIndifferentAccess, and my own spin-off from Ruby Facets, the Stash class. It's just one of those things that is genuinely useful. I am not partial to any particular implementation, but I trust Ara's skills and the short name "Map" is pretty nice too.

(Just a side note: Personally I'd prefer the functionality of Map in place of Hash itself, since it has the functionality people want 80% or more of the time, but I realize that is likely to have too many backward compatibility issues for real consideration.)
=end

History
#1 - 10/19/2010 11:30 AM - shyouhei (Shyouhei Urabe)
=begin
FYI: There has been a Mash merge request about a year ago. Please take a look at http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/26492 and its followups.
=end

#2 - 10/20/2010 09:44 PM - rklemme (Robert Klemme)
=begin
On Tue, Oct 19, 2010 at 4:34 AM, Shyouhei Urabe redmine@ruby-lang.org wrote:

Issue #3963 has been updated by Shyouhei Urabe.

FYI: There has been a Mash merge request about a year ago. Please take a look at http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/26492 and its followups.

http://redmine.ruby-lang.org/issues/show/3963

Especially http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/26513
I'd say. I think Matz hit the nail on the head here. After all, there are Symbol#to_s, String#to_s, Symbol#to_sym and String#to_sym which make conversion a nobrainer if it is needed.

Cheers

robert

--
remember.guy do |as, often| as.you_can - without end http://blog.rubybestpractices.com/
=end

#3 - 11/05/2010 03:19 AM - trans (Thomas Sawyer)
=begin
I can't check that post b/c service is down :(

But I would argue that a class like this goes much further than key conversions. One wants to prevent any other key types from being used at all.

(Also, I want follow up that Ara's Map class might not be the best choice. I think maybe he's added too many "extra features", like OpendStruct-like

03/18/2020
lookup/. Maybe Mash is a better implementation? But I do like the name Map.)

=end

#4 - 03/25/2012 02:24 PM - mame (Yusuke Endoh)
- Description updated
- Status changed from Open to Assigned
- Assignee set to akr (Akira Tanaka)

#5 - 03/25/2012 02:24 PM - akr (Akira Tanaka)
- Status changed from Assigned to Rejected

I think no hope to add Map class as builtin.

#6 - 03/27/2012 12:53 AM - mame (Yusuke Endoh)
Hello,

2012/3/25, akr (Akira Tanaka) akr@fsij.org:

I think no hope to add Map class as builtin.

I agree for two reasons.
The name "map" is not suitable. In computer science, "map" stands for
an abstract data structure that preserves correspondence between a key
Hash, binary tree, and "Ara Howard's Map" are all instances of map.
So the name is too general for the library.

In addition, standard libraries are basically shrinking. It is hard
to add a new library except a good reason.

--
Yusuke Endoh mame@tsg.ne.jp

#7 - 03/27/2012 01:49 AM - trans (Thomas Sawyer)
Just to be sure, I think I should clarify.

I do not mean Ruby ought provide Ara's Map class specifically (though, for the most part it is probably a very good implementation which is why I
reference it). But that Ruby provide at least some sort of "Hash with indifferent access" out of the box. In other words, a Hash class that only allows
for "text keys" (whether they are symbols or strings under the hood, doesn't matter).

The thing is, more often than not, that is the way we tend to use a Hash. For example, named parameters for methods optimally would use a Hash
that makes no distinction String vs Symbol keys.

I figured that a new class is the most likely to be accepted, but I will point out, this could be addressed in another way. If Hash itself supported a key
casting procedure, akin to the default value procedure, then we could do something like:

```ruby
h = Hash.new
h.convert_key{ |k| k.to_sym }

h['a'] = 1
h #=> [{a => 1}]
```

If we could do that, then there would be no need of an alternate class. And this would be a much more flexible approach too. But, as I said, I suspect
this has less chance of acceptance b/c I suspect it would be problematic implementation wise. For that reason I proposed having the a new class in
standard library.

#8 - 03/27/2012 02:20 AM - mame (Yusuke Endoh)
Did you read the thread Shyouhei said?

shyouhei wrote:

FYI: There has been a Mash merge request about a year ago. Please take a look at
[http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/26492](http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/26492)
and its followups.

I don't remember it in detail, but a similar idea you are talking about
was discussed, I guess. Please explain how your idea differs from the old discussion if any.

--
Yusuke Endoh mame@tsg.ne.jp

#9 - 03/27/2012 02:32 AM - trans (Thomas Sawyer)
Not much difference, except perhaps for the more flexible #convert_key idea. I was just clarifying in this ticket b/c of some of the comments made about Map specifically.

The conversation referenced did not seem to have any kind of conclusion. I see that matz indicated a preference for a possible "compare by string" directive. So maybe #convert_key is not such a bad notation as I thought?

#10 - 03/27/2012 02:42 AM - mame (Yusuke Endoh)
- Status changed from Rejected to Assigned
- Assignee changed from akr (Akira Tanaka) to matz (Yukihiro Matsumoto)

Okay, thanks. I mark this ticket as matz's issue.

--
Yusuke Endoh mame@tsg.ne.jp

#11 - 04/10/2012 06:51 PM - matz (Yukihiro Matsumoto)
- Status changed from Assigned to Rejected

I reject the original idea of Map inclusion to the core. Map and Hash are too similar.

Matz.