Ruby - Feature #5123

Alias Hash 1.9 as OrderedHash

07/31/2011 04:19 PM - alexeymuranov (Alexey Muranov)

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

Description

I read that it was a controversial decision to make Hash in Ruby 1.9 ordered.
It is not clear if the present implementation is the best possible.
I would like to express my thought.

It would be nice if the ordered Hash in Ruby 1.9 was aliased as OrderedHash.
That way people who rely on preserving the insertion order in some application (me, for example) could explicitly use OrderedHash, and developers of Ruby would be free to redefine Hash in future versions if a better implementation that a doubly-linked circular list is found.
(I read something about a possibility of using "red–black tree".)

Thanks.

Alexey Muranov.

History

#1 - 08/02/2011 04:32 PM - znz (Kazuhiro NISHIYAMA)

- Status changed from Open to Assigned

- Assignee set to matz (Yukihiro Matsumoto)

#2 - 08/02/2011 04:54 PM - adgar (Michael Edgar)

As a corollary, for a few months I had one portion of my code which used Sets, and relied on the fact that Sets in 1.9 retain their order because they in turn use a Hash. The only real way I could document this requirement was to put comments in capital letters in the relevant portions of code so that I didn't forget that I relied on this behavior.

If I had been creating "OrderedSets" (though this sounds too close to SortedSet despite meaning something different), and OrderedSets use an OrderedHash, then it would have been clear in the code that insertion order was preserved, especially for developers new to 1.9.

I like this idea.

#3 - 08/03/2011 10:29 AM - trans (Thomas Sawyer)

IWon't this break orderedhash gem?

In any case, now that Hash preserves order, why would anyone ever think about changing it back and break old code?

#4 - 08/03/2011 04:23 PM - yeban (Anurag Priyam)

> It would be nice if the ordered Hash in Ruby 1.9 was aliased as OrderedHash.
> That way people who rely on preserving the insertion order in some application (me, for example) could explicitly use OrderedHash, and developers of Ruby would be free to redefine Hash in future versions if a better implementation that a doubly-linked circular list is found.
> (I read something about a possibility of using "redâ€“black tree".)

This sounds like one of those variable names with a type prefixed to
it, like sequence_string, or sequence_array. They look ugly. It would
have mad more sense if Ruby were ever to ship with two different Hash
implementations. Besides, the aliasing can be done at the application
level too. So if the Hash implementation changes tomorrow, you are
free to re-alias to some other ordered hash implementation.

--
Anurag Priyam
http://about.me/yeban/

#5 - 08/07/2011 07:23 AM - alexeymuranov (Alexey Muranov)

Thomas Sawyer wrote:

> lWon't this break orderedhash gem?
>
> In any case, now that Hash preserves order, why would anyone ever think about changing it back and break old code?

To improve performance, for example, or to restore the original meaning of "hash".

Alexey Muranov.

#6 - 08/07/2011 07:24 AM - alexeymuranov (Alexey Muranov)

Anurag Priyam wrote:

> Besides, the aliasing can be done at the application
> level too. So if the Hash implementation changes tomorrow, you are
> free to re-alias to some other ordered hash implementation.

But then i will have to edit my code to upgrade to a new version of Ruby.

Alexey Muranov.

Update 2011-11-09. "It would have made more sense if Ruby were ever to ship with two different Hash implementations." -- Anurag Priyam
Why not? Implementing Hash and OrderedHash separately some day (50 years from now) may gain some speed for pure Hash.

Update 2011-12-04. This is where i first read about red-black trees for hash: http://www.igvita.com/2009/02/04/ruby-19-internals-ordered-hash/

Update 2012-05-04. I still like my original proposal because the use of OrderedHash constant documents the behavior, but if it is not accepted, another way to use unordered "faster" hashes would be to call them OptimizedHash, and have a method Hash#optimized for conversion:

```
    a = {}.optimized
```

With the original proposal, the method would be Hash#ordered:

```
    b = {}.ordered
```

#7 - 10/28/2012 12:00 AM - yhara (Yutaka HARA)

- Category set to core

- Target version set to 2.6

#8 - 11/01/2012 05:09 AM - headius (Charles Nutter)

The decision is already made. Hash has been ordered since 1.9.1 (or 1.9.2?) and changing it now would break all code that expects ordering. Ordering also does not necessarily limit options for implementation...it just makes certain implementations less efficient.

There's a separate bug still being evaluated to add RBTree to core...perhaps that would fulfill what you need?

#9 - 11/01/2012 05:29 AM - alexeymuranov (Alexey Muranov)

I was only proposing an alias for now. Maybe OHash, OSet?

#10 - 12/19/2015 11:43 AM - alexeymuranov (Alexey Muranov)

I have just stumbled upon this: the Immutable collections for JavaScript has both Map and OrderedMap.

One more: in Haskell, there are containers and unordered-containers.

#11 - 02/20/2018 08:19 AM - matz (Yukihiro Matsumoto)

- Status changed from Assigned to Rejected

Today, everyone knows Ruby's Hash is a OrderedHash.

Matz.