Ruby - Bug #13855

Hash#compact! returns nil if the hash is empty

08/31/2017 03:24 PM - elandesign (Paul Smith)

Status:	Closed		
Priority:	Normal		
Assignee:	nobu (Nobuyoshi Nakada)		
Target version:			
ruby -v:	ruby 2.4.1p111 (2017-03-22 revision 58053) [x86_64-darwin16]	**Backport:**	2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: UNKNOWN

Description

This behaviour feels like a bug to me.

From the documentation (with my emphasis):

> compact! → hsh
> Removes all nil values from the hash. **Returns the hash**.

However if the hash contains no keys, the method returns nil.

```
irb(main):001:0> {}.compact!
=> nil

# For Comparison
irb(main):002:0> { foo: nil }.compact!
=> {}
irb(main):003:0> {}.compact
=> {}
irb(main):004:0> { foo: nil }.compact
=> {}
```

Associated revisions

Revision 0118b985459ad20520af09e5202774b3cc52b491 - 09/02/2017 01:08 AM - nobu (Nobuyoshi Nakada)

Update Hash#compact! documentation [ci skip]

- hash.c (rb_hash_compact_bang): [DOC] update the case if no changes were made. [ruby-core:82591] [Bug #13855] [Fix GH-1692]

Author: Lucas Buchala lucasbuchala@gmail.com

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@59719 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 0118b985 - 09/02/2017 01:08 AM - nobu (Nobuyoshi Nakada)

Update Hash#compact! documentation [ci skip]

- hash.c (rb_hash_compact_bang): [DOC] update the case if no changes were made. [ruby-core:82591] [Bug #13855] [Fix GH-1692]

Author: Lucas Buchala lucasbuchala@gmail.com

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@59719 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 08/31/2017 04:45 PM - lucasbuchala (Lucas Buchala)

Rather than a bug, I wonder if this is just a documentation omission.

In case of a documentation omission, I created a PR:

https://github.com/ruby/ruby/pull/1692

#2 - 08/31/2017 06:44 PM - shevegen (Robert A. Heiler)

Hmm.

I had a look at class String and class Array what they do.

First Array:

```
array = [1,2,3] # => [1, 2, 3]
array.compact!  # => nil
```

Next class String - it has no .compact but perhaps we can use
.delete! which may be somewhat similar:

```
string = 'abc' # => "abc"
string.delete! 'd' # => nil
```

So I assume that the nil returned there may be consistent behaviour. Probably
by nature of the assumption of what is returned, modification in place on
the same object, or a modification what would return a new object.

However had I do agree that, either way, the documentation could be more explicit.

No harm in mentioning that the hash.compact! variant will return nil, which is
what it is presently doing (for an empty hash).

For non-empty hashes as result, it indeed returns the remaining hash, which is
a bit strange, since it may also return a nil ... so people have to check whether
they really have a hash there or just an il:

```
hash = { foo: nil, bar: 'yes' } # => {:foo=>nil, :bar=>"yes"}
hash.compact!                   # => {:bar=>"yes"}
```

I do however had also agree that the behaviour may be a bit unexpected... so perhaps
the documentation can be fixed either way and then we can ask whether the behaviour
is the way it should be - but I actually assume that it may be just a documentation
problem.

#3 - 08/31/2017 10:21 PM - elandesign (Paul Smith)

lucasbuchala (Lucas Buchala) wrote:

> Rather than a bug, I wonder if this is just a documentation omission.
>
> In case of a documentation omission, I created a PR:
>
> https://github.com/ruby/ruby/pull/1692

Oh that's interesting - I hadn't noticed that returning nil if the object was unchanged was the actual behaviour there. I agree it could be seen as a documentation issue - honestly, I'd be happier with the bang method returning the affected object, but so long as the behaviour is expected either way works for me.

#4 - 09/01/2017 02:28 AM - duerst (Martin Dürst)

- Assignee set to nobu (Nobuyoshi Nakada)

nil is returned if there is no change, not only if the hash is empty:

```
hash = { foo: :bar, one: :two } # => {:foo=>:bar, :one=>:two}
hash.compact!                   # => nil
```

This is consistent with many other bang methods. The idea is that the result is available in the changed variable already, and the return value can be used in an if or while expression.

The patch looks good, I hope Nobu can apply it (I can't automatically integrate patches from github).

#5 - 09/02/2017 01:08 AM - nobu (Nobuyoshi Nakada)

- Status changed from Open to Closed

Applied in changeset trunk|r59719.

Update Hash#compact! documentation [ci skip]

- hash.c (rb_hash_compact_bang): [DOC] update the case if no changes were made. [ruby-core:82591] [Bug #13855] [Fix GH-1692]

Author: Lucas Buchala lucasbuchala@gmail.com