Ruby master - Feature #8707

Hash#reverse_each

07/30/2013 10:58 PM - Glass_saga (Masaki Matsushita)

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

Description

Currently, {}.reverse_each calls Enumerable#reverse_each.
It will make array and its size can be large.
I made Hash#reverse_each to avoid array creation and performance improvement.

benchmark:

```
require "benchmark"

Size = 10000
HASH = Hash[*Array.new(Size) {|i| [i, true] }.flatten]

Benchmark.bmbm do |x|
x.report("Hash#reverse_each") do
300.times do
HASH.reverse_each {|a, b|}
end
end
end
```

result:

```
trunk(r42256):
Rehearsal ----------------------------------------------------
Hash#reverse_each   1.210000   0.000000   1.210000 (  1.207964)
------------------------------------------- total: 1.210000sec

                        user     system      total        real

Hash#reverse_each   0.950000   0.000000   0.950000 (  0.951069)

proposal:
Rehearsal ----------------------------------------------------
Hash#reverse_each   0.600000   0.000000   0.600000 (  0.600242)
------------------------------------------- total: 0.600000sec

                        user     system      total        real

Hash#reverse_each   0.450000   0.000000   0.450000 (  0.459006)
```

History

#1 - 07/31/2013 12:48 PM - matz (Yukihiro Matsumoto)

- Status changed from Open to Feedback

Do we really need it? What is use-cases?

Matz.

#2 - 07/31/2013 02:29 PM - Anonymous

Matz: This is quite a significant performance improvement and therefore I think it is worthwhile.

On 30/07/2013, at 20:48, "matz (Yukihiro Matsumoto)" matz@ruby-lang.org wrote:

Issue #8707 has been updated by matz (Yukihiro Matsumoto).

Status changed from Open to Feedback

Do we really need it? What is use-cases?

Matz.

Feature #8707: Hash#reverse_each
https://bugs.ruby-lang.org/issues/8707#change-40769

Author: Glass_saga (Masaki Matsushita)
Status: Feedback
Priority: Normal
Assignee:
Category: core
Target version: current: 2.1.0

Currently, {}.reverse_each calls Enumerable#reverse_each.
It will make array and its size can be large.
I made Hash#reverse_each to avoid array creation and performance improvement.

benchmark:

```
require "benchmark"

Size = 10000
HASH = Hash[*Array.new(Size) {|i| [i, true] }.flatten]

Benchmark.bmbm do |x|
x.report("Hash#reverse_each") do
300.times do
HASH.reverse_each {|a, b|}
end
end
end
```

result:

```
trunk(r42256):
Rehearsal ----------------------------------------------------
Hash#reverse_each   1.210000   0.000000   1.210000 (  1.207964)
------------------------------------------- total: 1.210000sec

                        user     system      total        real

Hash#reverse_each   0.950000   0.000000   0.950000 (  0.951069)

proposal:
Rehearsal ----------------------------------------------------
Hash#reverse_each   0.600000   0.000000   0.600000 (  0.600242)
------------------------------------------- total: 0.600000sec

                        user     system      total        real

Hash#reverse_each   0.450000   0.000000   0.450000 (  0.459006)
```

--
http://bugs.ruby-lang.org/

#3 - 07/31/2013 04:23 PM - duerst (Martin Dürst)

Hello Charlie,

On 2013/07/31 14:24, Charlie Somerville wrote:

> Matz: This is quite a significant performance improvement and therefore I think it is worthwhile.

Only if the new method is actually used in practice.
That's why Matz is asking for use cases.

Regards, Martin.

On 30/07/2013, at 20:48, "matz (Yukihiro Matsumoto)"matz@ruby-lang.org wrote:

Issue #8707 has been updated by matz (Yukihiro Matsumoto).

Status changed from Open to Feedback

Do we really need it? What is use-cases?

Matz.

Feature #8707: Hash#reverse_each
https://bugs.ruby-lang.org/issues/8707#change-40769

Author: Glass_saga (Masaki Matsushita)
Status: Feedback
Priority: Normal
Assignee:
Category: core
Target version: current: 2.1.0

Currently, {}.reverse_each calls Enumerable#reverse_each.
It will make array and its size can be large.
I made Hash#reverse_each to avoid array creation and performance improvement.

benchmark:

```
require "benchmark"

Size = 10000
HASH = Hash[*Array.new(Size) {|i| [i, true] }.flatten]

Benchmark.bmbm do |x|
x.report("Hash#reverse_each") do
300.times do
HASH.reverse_each {|a, b|}
end
end
end
```

result:

```
trunk(r42256):
Rehearsal ----------------------------------------------------
Hash#reverse_each   1.210000   0.000000   1.210000 (  1.207964)
------------------------------------------- total: 1.210000sec

                        user     system      total        real

Hash#reverse_each   0.950000   0.000000   0.950000 (  0.951069)

proposal:
Rehearsal ----------------------------------------------------
Hash#reverse_each   0.600000   0.000000   0.600000 (  0.600242)
------------------------------------------- total: 0.600000sec

                        user     system      total        real

Hash#reverse_each   0.450000   0.000000   0.450000 (  0.459006)
```

--
http://bugs.ruby-lang.org/

#4 - 08/09/2013 07:34 PM - ko1 (Koichi Sasada)

- Assignee set to matz (Yukihiro Matsumoto)

#5 - 08/10/2013 06:19 PM - funny_falcon (Yura Sokolov)

Hash in Ruby1.9+ is hash table + double linked list - this is classic structure for LRU cache.

Simple LRU cache could be build with:

```
class LRU
  def initialize
    @hash = {}
  end

  def []=(k, v)
    @hash.delete k
    @hash[k] = v
  end

  def [](k)
    if v = @hash.delete k
      @hash[k] = v
    end
    v
  end

  def first
    @hash.first
  end

  def shift
    @hash.shift
  end

  # saves tail items until first stale item signaled
  def drop_stale
    save = true
    stale = []
    @hash.reverse_each{|k, v|
      unless save && yield(k, v)
        save = false
        v = @hash.delete k
        stale << [k, v]
      end
    }
    stale
  end
end
```

So that, reverse_each is very critical to be fast at this point

#6 - 01/30/2014 06:17 AM - hsbt (Hiroshi SHIBATA)

- Target version changed from 2.1.0 to 2.2.0

#7 - 04/11/2014 06:24 PM - trans (Thomas Sawyer)

Can #reverse be an Enumerator?

#8 - 01/05/2018 09:00 PM - naruse (Yui NARUSE)

- Target version deleted (2.2.0)

Files

patch.diff	7.55 KB	07/30/2013	Glass_saga (Masaki Matsushita)