

Ruby trunk - Feature #6641

Hash.auto constructor

06/25/2012 09:34 AM - trans (Thomas Sawyer)

Status: Open	
Priority: Normal	
Assignee: matz (Yukihiro Matsumoto)	
Target version:	
Description	
<pre>=begin It is not uncommon to need a Hash following the pattern: Hash.new{ h,k h[k]={} } Another common example: Hash.new{ h,k h[k]=[] } This is common enough that it would very nice if we could have a more concise form, e.g. Hash.auto{ {} } Or for the second example: Hash.auto{ [] } Pure Ruby implementation is pretty simple: def Hash.auto Hash.new{ h,k h[k] = yield } end I think it would be nice to have in Core. This is secondary, but it just occurred to me. Could there even be a literal notation for the above? Something like: {}{[]} ? =end</pre>	

History

#1 - 06/25/2012 07:33 PM - Eregon (Benoit Daloze)

trans (Thomas Sawyer) wrote:

It is not uncommon to need a Hash following the pattern:

```
Hash.new{ |h,k| h[k]=[] }
```

This is common enough that it would very nice if we could have a more concise form, e.g.

```
Hash.auto{ [] }
```

I agree, this is a pattern I see quite often and the `Hash.new { |h,k| h[k] = ... }` form is not the most natural or expressive in my opinion.

But at the same time, this is clearly a specialization of `Hash.new` with a block, which loses some flexibility (you can not use the key for example). I'm curious what others think of it.

#2 - 06/25/2012 09:24 PM - rosenfeld (Rodrigo Rosenfeld Rosas)

Maybe something like `Hash.with_default{[]}`. "with_default" could be both a class and an instance method so that you could have something like `{a: 1, b: 2}.with_default(0).merge...` It would always return the hash itself...

Groovy has something like this (not quite the same behavior). In Groovy you can define default values as `[:].withDefault{0}`. You can optionally use the

block argument as the key.

So I guess it would be great if Ruby with_default supported all those syntax below:

```
Hash.with_default(0)
Hash.with_default{Time.now}
Hash.with_default{|k| k.to_sym}
{}.with_default(0)
{}.with_default{Time.now}
{}.with_default{|k| k.to_sym}
```

#3 - 06/25/2012 09:29 PM - jwille (Jens Wille)

Eregon (Benoit Daloz) [2012-06-25 12:33]:

But at the same time, this is clearly a specialization of Hash.new with a block, which loses some flexibility (you can not use the key for example).
why not? you can yield the key to the block.

I'm curious what others think of it.
i like it and i use it occasionally:

<http://blackwinter.github.com/ruby-nuggets/Nuggets/Hash/NestMixin.html>
https://github.com/blackwinter/ruby-nuggets/blob/master/lib/nuggets/hash/nest_mixin.rb
https://github.com/blackwinter/ruby-nuggets/blob/master/spec/nuggets/hash/nest_spec.rb

#4 - 06/26/2012 12:19 PM - nobu (Nobuyoshi Nakada)

-1.
I don't like the destructive default proc, which sets new keys automatically.

#5 - 06/26/2012 02:23 PM - kosaki (Motohiro KOSAKI)

```
Hash.with_default(0)
Hash.with_default{Time.now}
Hash.with_default{|k| k.to_sym}
{}.with_default(0)
{}.with_default{Time.now}
{}.with_default{|k| k.to_sym}
```

I like with_default than auto. It describe the behavior more preciously.
auto is too vague to me.

#6 - 10/25/2012 07:38 PM - yhara (Yutaka HARA)

- Target version changed from 2.0.0 to 2.6

#7 - 02/07/2013 08:21 AM - phluid61 (Matthew Kerwin)

I agree with nobu-shi, if I were to encounter Hash::with_default &block I would assume it was the equivalent of:

```
Hash.new{ |h,k| yield }
```

rather than:

```
Hash.new{ |h,k| h[k] = yield }
```

Perhaps there could be ::with_default and ::with_default!

#8 - 02/22/2013 09:08 AM - ko1 (Koichi Sasada)

- Assignee set to matz (Yukihiro Matsumoto)

#9 - 12/25/2017 06:15 PM - naruse (Yui NARUSE)

- Target version deleted (2.6)