Ruby master - Feature #13179

Deep Hash Update Method

02/01/2017 06:46 PM - bettisworth (wurde _)

Status:	Rejected	
Priority:	Normal	
Assignee:		
Target version:		

Description

I came across a scenario where I needed the ability to update a deeply nested hash (Rails i18n yaml files). This seemed like something that would exist naturally in the DSL of ruby, but I could only find dig() method which only retrieves values if they exist. 24 hours later I wrote a update_deep_hash method (I wouldn't wish this type of recursive coding on anyone else within the Ruby community).

Attached is the solution I hacked together. My question is if we can have a Hash.dig() method which reaches into a deep hash can we expand on this to include update methods of similar nature?

Related issues:

Is duplicate of Ruby master - Feature #11747: "bury" feature, similar to 'dig...	**Rejected**

History

#1 - 02/02/2017 06:50 AM - nobu (Nobuyoshi Nakada)

- Status changed from Open to Feedback

- Tracker changed from Bug to Feature

dig is not only for Hash but also for Array, Struct, and OpenStruct.
What objects would you expect as intermediate containers if not present?

Regarding the name, bury occurred to me.
Hash#dig= might fit if obj.method(arg,...) = val syntax were possible.

#2 - 02/03/2017 04:22 AM - shevegen (Robert A. Heiler)

> Regarding the name, bury occurred to me.

.dig and .bury - I can already feel on Halloween the .zombies() coming up!

#3 - 02/03/2017 05:05 AM - bettisworth (wurde _)

Any method name four or less characters long is a big win. Using bury would be much nicer than my current update_deep_hash calls.

I wasn't aware of dig in the other classes. I may be overlooking something but why not just use the class dig is used on as a container? Another way would be to allow setting the default container via an optional argument like when setting default values for a Hash, Array, and so on.

#4 - 02/23/2017 06:48 AM - nobu (Nobuyoshi Nakada)

- Is duplicate of Feature #11747: "bury" feature, similar to 'dig' but opposite added

#5 - 03/02/2017 04:56 AM - shyouhei (Shyouhei Urabe)

This is possible in perl:

```
use strict;
use warnings;
use Data::Dumper qw(Dumper);

my %hash = qw();
$hash{q}{w}{e}{r}{t} = 'y';

warn(Dumper(\%hash));
```

From perl's POV it seems ruby lacks this feature.

#6 - 03/13/2017 09:21 AM - matz (Yukihiro Matsumoto)

- Status changed from Feedback to Rejected

It's no difference from #11747. You have to come up with a better name candidate and concrete use-case.

Matz.

#7 - 01/10/2018 01:51 PM - MarioRuiz (Mario Ruiz Sánchez)

bettisworth (wurde _) wrote:

> I came across a scenario where I needed the ability to update a deeply nested hash (Rails i18n yaml files). This seemed like something that would exist naturally in the DSL of ruby, but I could only find dig() method which only retrieves values if they exist. 24 hours later I wrote a update_deep_hash method (I wouldn't wish this type of recursive coding on anyone else within the Ruby community).
>
> Attached is the solution I hacked together. My question is if we can have a Hash.dig() method which reaches into a deep hash can we expand on this to include update methods of similar nature?

What I am using in my code is a simple bury method... at least for the best cases is working nice:

```
#my_hash.bury([:accent,2,:original],"the value to set")
class Hash
  def bury(where, value)
      me=self
      where[0..-2].each{|key|
        me=me[key]
      }
      me[where[-1]]=value
  end
end

#my_array.bury([2,1,:original],"the value to set")
class Array
  def bury(where, value)
      me=self
      where[0..-2].each{|key|
        me=me[key]
      }
      me[where[-1]]=value
  end
end
```

#8 - 07/14/2019 01:29 PM - bkatzung (Brian Katzung)

Gem XKeys, since 2014.

```
require 'xkeys'
h = {}.extend XKeys::Auto # auto Hash/Array
h[:accent, 2, :original] = "the value to set"
# {:accent=>[nil, nil, {:original=>"the value to set"}]}

h = {}.extend XKeys::Hash # Hash only
h[:accent,2,:original]='the value to set'
# {:accent=>{2=>{:original=>"the value to set"}}}

a = [].extend XKeys::Auto
a[2, 1, :original] = "the value to set"
# [nil, nil, [nil, {:original=>"the value to set"}]]
a[2, 1] # (slice 2, 1) [[nil, {:original=>"the value to set"}]]
a[2, 1, {}] # (a[2][1]) {:original=>"the value to set"}
```

Files

update_deep_hash.rb	2.26 KB	02/01/2017	bettisworth (wurde _)