Hi,

I saw a bit strange behavior (at least for me) with the code below; an object given as an argument to a method with a default value and a keyword argument becomes Hash class if the object has 'to_hash' method with ruby >= 2.2, but it is not converted to Hash with ruby 2.1.

I expected the behavior of ruby 2.1.

Is this a bug or expected behavior of ruby >= 2.2?

I tested both on Linux x86_64 (CentOS 6) and Mac OS X 10.11.6.

```ruby
class Test1
end

class Test2
  def to_hash
    {0 => [0, 1]}
  end
end

def argtest1 foo, bar: nil
  p foo.class
end

def argtest2 foo = nil, bar: nil
  p foo.class
end

test1 = Test1.new

test2 = Test2.new

argtest1(test1) # => Test1
argtest1(test2) # => Test2

argtest2(test1) # => Test1
argtest2(test2) # => Hash (2.2.5p319, 2.3.1p112, 2.4.0-preview2), Test2 (2.1.10p492)
```

Regards,
Hiroyuki Tanaka

Related issues:
- Related to Ruby master - Feature #14183: "Real" keyword argument
  - Closed
- Has duplicate Ruby master - Bug #12884: Using a HashWithIndifferentAccess wit...  - Closed

History
#1 - 10/13/2016 07:25 PM - kernigh (George Koehler)

It's not what I expected, but Ruby is splitting test2 into two hashes, with all keywords into the second hash. There are no keywords in {0 => [0, 1]}, so Ruby passes {0 => [0, 1]} as foo and {} as keywords.
argtest2(0 => [0, 1], bar: 2) # => foo = {0=>[0, 1]} (Hash), bar: 2
argtest2(bar: 2, 0 => [0, 1]) # => foo = {0=>[0, 1]} (Hash), bar: 2
argtest2(0 => [0, 1]) # => foo = {0=>[0, 1]} (Hash), bar: nil

So argtest2(test2) acts like argtest2(0 => [0, 1]). I don't know why Ruby does this split, or why Ruby 2.1 was different.

#2 - 10/31/2016 04:45 AM - shyouhei (Shyouhei Urabe)
- Has duplicate Bug #12884: Using a HashWithIndifferentAccess with a default value in a function with a keyword parameter converts it to a Hash. added

#3 - 10/31/2016 06:01 AM - nobu (Nobuyoshi Nakada)
- Description updated
- Subject changed from Object converted to Hash unexpectedly under certain method call to Object converted to Hash unexpectedly under certain method call

#4 - 10/31/2016 03:04 PM - rrroybbbean (RRRoy BBBean)
I get the same odd behavior with:
ruby --version ruby 2.3.1p112 (2016-04-26 revision 54768) [x86_64-linux]
ruby --version ruby 2.4.0-preview2 (2016-09-09 trunk 56129) [x86_64-linux]

It looks like the combination ( foo = nil, bar: nil ) causes the issue.

On 10/31/2016 01:01 AM, nobu@ruby-lang.org wrote:

Issue #12821 has been updated by Nobuyoshi Nakada.
Subject changed from Object converted to Hash unexpectedly under certain method call to Object converted to Hash unexpectedly under certain method call
Description updated

Bug #12821: Object converted to Hash unexpectedly under certain method call
https://bugs.ruby-lang.org/issues/12821#change-61140

- Author: Hiroyuki Tanaka
- Status: Open
- Priority: Normal
- Assignee:
- ruby -v: 2.2.5p319, 2.3.1p112, 2.4.0-preview2

* Backport: 2.1: UNKNOWN, 2.2: UNKNOWN, 2.3: UNKNOWN

Hi,

I saw a bit strange behavior (at least for me) with the code below: an object given as an argument to a method with a default value and a keyword argument becomes Hash class if the object has 'to_hash' method with ruby >= 2.2, but it is not converted to Hash with ruby 2.1. I expected the behavior of ruby 2.1.

Is this a bug or expected behavior of ruby => 2.2?

I tested both on Linux x86_64 (CentOS 6) and Mac OS X 10.11.6.

class Test1
  end

class Test2
  def to_hash
    { 0 => [0, 1] }
  end
end

def argtest1 foo, bar: nil
  p foo.class
end

def argtest2 foo = nil, bar: nil
  p foo.class
end

test1 = Test1.new
test2 = Test2.new

argtest1(test1)  # => Test1
argtest1(test2)  # => Test2

argtest2(test1)  # => Test1
argtest2(test2)  # => Hash (2.2.5p319, 2.3.1p112, 2.4.0-preview2), Test2 (2.1.10p492)

Regards,
Hiroyuki Tanaka

#5 - 03/11/2017 05:12 PM - naruse (Yui NARUSE)
- Backport deleted (2.1: UNKNOWN, 2.2: UNKNOWN, 2.3: UNKNOWN)

#6 - 03/11/2017 05:12 PM - naruse (Yui NARUSE)
- Status changed from Open to Closed

#7 - 12/14/2017 07:24 AM - hsbt (Hiroshi SHIBATA)
- Related to Feature #14183: "Real" keyword argument added