There are two ways to find out what fields an OpenStruct instance has. One is through \texttt{inspect}, however that returns a String that needs to be parsed.

The second is by white box engineering, looking at OpenStructs source code and seeing that in fact, it has a hash and getting the keys of that hash...

The second way is faster, more robust, but will break once OpenStruct will be re-engineered...

So I suggest to add an explicit method to return a list of fields in an OpenStruct instance:

\begin{verbatim}
--- ostruct.rb.old 2009-04-23 15:26:45.000000000 +0200
+++ ostruct.rb 2009-04-23 15:32:41.000000000 +0200
@@ -110,6 +110,15 @@
    @table.delete name.to_sym
end

-  #
-  # Returns an Array containing the fields of an OpenStruct instance
-  #
-  # p record.fields # -> [:age, :pension, :name]
-  #
-  def fields
-  @table.keys
+
-  InspectKey = :inspect_key # :nodoc:

  #
  =end
\end{verbatim}

\textbf{Related issues:}

- Related to Ruby master - Feature #6056: Enhancements to OpenStruct \textcolor{red}{Closed 02/21/2012}
- Has duplicate Ruby master - Feature #5008: Equal rights for Hash (like Array,... \textcolor{red}{Rejected 07/10/2011}

\textbf{Associated revisions}

\texttt{Revision db66739c - 04/16/2012 03:16 AM - marcandre (Marc-Andre Lafortune)}

- \texttt{lib/ostruct.rb: Add OpenStruct#to_h [Feature #6276] [ref #1400] [rubyspec:9e0250b2fc6f]}
- \texttt{git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@35342 b2dd03c8-39d4-4d8f-98ff-823fe69b080e}

\texttt{Revision 35342 - 04/16/2012 03:16 AM - marcandre (Marc-Andre Lafortune)}

- \texttt{lib/ostruct.rb: Add OpenStruct#to_h [Feature #6276] [ref #1400] [rubyspec:9e0250b2fc6f]}

\texttt{Revision 35342 - 04/16/2012 03:16 AM - marcandre (Marc-Andre Lafortune)}
lib/ostruct.rb: Add OpenStruct#to_h [Feature #6276] [ref #1400] [rubyspec:9e0250b2fc6f]

Revision 35342 - 04/16/2012 03:16 AM - marcandre (Marc-Andre Lafortune)

lib/ostruct.rb: Add OpenStruct#to_h [Feature #6276] [ref #1400] [rubyspec:9e0250b2fc6f]

Revision 35342 - 04/16/2012 03:16 AM - marcandre (Marc-Andre Lafortune)

lib/ostruct.rb: Add OpenStruct#to_h [Feature #6276] [ref #1400] [rubyspec:9e0250b2fc6f]

Revision 35342 - 04/16/2012 03:16 AM - marcandre (Marc-Andre Lafortune)

lib/ostruct.rb: Add OpenStruct#to_h [Feature #6276] [ref #1400] [rubyspec:9e0250b2fc6f]

Revision 35342 - 04/16/2012 03:16 AM - marcandre (Marc-Andre Lafortune)

lib/ostruct.rb: Add OpenStruct#to_h [Feature #6276] [ref #1400] [rubyspec:9e0250b2fc6f]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- lib/ostruct.rb (each_pair): Add #each_pair [#1400]

Revision 37372 - 10/28/2012 09:18 PM - marcandre (Marc-Andre Lafortune)

- Category set to lib

=begin
=end

#2 - 09/19/2009 05:55 AM - marcandre (Marc-Andre Lafortune)

- Category set to lib

=begin
I believe this could be useful.

#members might be a better name, because it is similar in role to Struct.members

The problem is that any new method is a potential compatibility break. #members ?
=end

#3 - 03/17/2010 05:39 AM - DanRathbun (Dan Rathbun)

=begin
This can be done right now. In several ways.

```ruby
# obj is an class OpenStruct with some number of members.
ary = obj.methods(false).sort -> Array of attr getters and setters
keys=[]
ary.each { |i| keys.push(i) unless i.include?('=') }
# keys is now an Array of fieldnames
```

Also using existing method .marshal_dump which is really an attribute getter for @table (having a name different than the attribute.)

```
Just noticed .table as a attribute getter alias of .marshal_dump, BUT .table is set protected, while .marshal_dump is not?
```

Why is THAT?

```
# obj is an class OpenStruct with some number of members.
obj.marshal_dump.keys -> Array of keys
obj.marshal_dump.values -> Array of values
obj.marshal_dump.each { |k,v| ... }
```

The danger in accessing the Hash directly is someone's going to modify it without removing the attr getter and setter methods defined in the class instance. Which by the way is a bug in the current OpenStruct. It's delete_field method does just that, leaving the accessor methods without a key/value pair in @table.

There is also an opposite bug. IF an attempt is made, to create a field who's name is already used as a method (without the "=");

```
new_ostruct_member does NOT create the accessor methods, BUT does NOT return a indicator of success, so method_missing just goes ahead and adds the field to @table.
```

Ex:

```
obj.inspect='Sherlock Holmes'
```

The unless block (line 72) in new_ostruct_member needs an else clause thats raises a NameError Exception "#{name} is already in use as a method of #{self}.
```

Additionally, .marshal_load is flawed. It wipes out @table instead of appending; and puts everything in @table before checking it. The arg x should be typechecked as an OpenStruct or Hash; it should be put first in a temp reference. Looks like another argument may be needed for overwrite/ignore of matching keys.

=end

#4 - 04/04/2010 01:11 AM - znz (Kazuhiro NISHIYAMA)
- Category set to lib
- Target version set to 2.0.0

=begin

=end

#5 - 06/26/2011 01:59 PM - naruse (Yui NARUSE)
- Status changed from Open to Assigned
- Assignee set to matz (Yukihiro Matsumoto)

#6 - 02/14/2012 12:07 AM - mame (Yusuke Endoh)
- Assignee changed from matz (Yukihiro Matsumoto) to marcandre (Marc-Andre Lafortune)

Hello,

There is no maintainer for ostruct.
Marc-Andre, are you willing to be a maintainer and to commit your patch? I give +1 for members.

I consider Dan's comment is not objection. What Tomas wants is a way to get field names without depending on the openstruct internal. But the ways that Dan proposed are deeply depending on it. For other topics, please register bug ticket for each.

--
Yusuke Endoh mame@tsg.ne.jp

#7 - 02/14/2012 12:29 AM - peter_v (Peter Vandenabeele)
On Mon, Feb 13, 2012 at 4:08 PM, Yusuke Endoh mame@tsg.ne.jp wrote:

Issue #1400 has been updated by Yusuke Endoh.

Assignee changed from Yukihiro Matsumoto to Marc-Andre Lafortune
Hello,

There is no maintainer for ostruct. Marc-Andre, are you willing to be a maintainer and to commit your patch? I give +1 for members.

I consider Dan’s comment is not objection. What Tomas wants is a way to get field names without depending on the openstruct internal. But the ways that Dan proposed are deeply depending on it. For other topics, please register bug ticket for each.

I use OpenStruct#marshal_dump for this purpose, which is a public method and returns a clean hash.

$ irb
1.9.3p0 :001 > require 'ostruct'
=> false
1.9.3p0 :002 > car = OpenStruct.new
=> #
1.9.3p0 :003 > car.wheels = 4
=> 4
1.9.3p0 :004 > car.seats = 5
=> 5
1.9.3p0 :005 > car.marshal_dump
=> [{:wheels=>4, :seats=>5}]

Sincerely,
Peter

--
*** Available for a new project ***
Peter Vandenabeele
http://twitter.com/peter_v
http://rails.vandenabeele.com
http://coderwall.com/peter_v

#8 - 02/14/2012 12:53 AM - mame (Yusuke Endoh)
Hello,

2012/2/14 Peter Vandenabeele peter@vandenabeele.com:

I use OpenStruct#marshal_dump for this purpose, which is a public method and returns a clean hash.

Cleary, marshal_dump is only intended for Marshal. I don't know what version policy is applied to marshal format of a standard library, but I do NOT recommend using marshal_dump for the purpose. I bet it is not guaranteed.

--
Yusuke Endoh mame@tsg.ne.jp

#9 - 02/14/2012 12:53 AM - peter_v (Peter Vandenabeele)
On Mon, Feb 13, 2012 at 4:37 PM, Yusuke Endoh mame@tsg.ne.jp wrote:

Hello,

2012/2/14 Peter Vandenabeele peter@vandenabeele.com:

I use OpenStruct#marshal_dump for this purpose, which is a public method and returns a clean hash.

Cleary, marshal_dump is only intended for Marshal. I don't know what version policy is applied to marshal format of a standard library, but I do NOT recommend using marshal_dump for the purpose. I bet it is not guaranteed.
Many thanks for the clarification.

Peter

#10 - 02/14/2012 03:13 PM - marcandre (Marc-Andre Lafortune)
Hi,

Yusuke Endoh wrote:

> There is no maintainer for ostruct.
> Marc-Andre, are you willing to be a maintainer and to commit your patch? I give +1 for members.

With pleasure. I'll commit it tomorrow.

> I consider Dan's comment is not objection. What Tomas wants is a way to get field names without depending on the openstruct internal. But the ways that Dan proposed are deeply depending on it.

Agreed.

Marc-André

#11 - 02/15/2012 12:56 PM - marcandre (Marc-Andre Lafortune)
After reviewing the library, I'm thinking it could be more useful to implement instead each_pair that would yield keys with the corresponding value (or return an enumerator if no block given).

*) It matches the equivalent Struct#each_pair
*) It's more powerful
*) It's highly unlikely to generate a conflict

Thoughts?

#12 - 02/15/2012 07:53 PM - peter_v (Peter Vandenabeele)
On Wed, Feb 15, 2012 at 4:56 AM, Marc-Andre Lafortune <ruby-core@marc-andre.ca> wrote:

Issue #1400 has been updated by Marc-Andre Lafortune.

After reviewing the library, I'm thinking it could be more useful to implement instead each_pair that would yield keys with the corresponding value (or return an enumerator if no block given).

*) It matches the equivalent Struct#each_pair
*) It's more powerful
*) It's highly unlikely to generate a conflict

Thoughts?

Seems good.

Is there a specified order for the enumeration? (I presume not, but curious).

Peter

#13 - 02/22/2012 04:00 AM - trans (Thomas Sawyer)
If #each_pair, why not #each? I realize it's an exception to the fields that can be used, but since a few of those are inevitable no matter what, it seems like an acceptable one.

#14 - 02/22/2012 04:02 AM - trans (Thomas Sawyer)
It was also recommended to me to suggest #to_h here.

```ruby
def to_h
  @table.dup
end```

09/25/2021
It would be a much easier way of working with the underlying table, such as getting field names.

`openstruct.to_h.keys`

And of course much more.

---

**#15 - 03/26/2012 08:33 AM - marcandre (Marc-Andre Lafortune)**

Hi,

Peter Vandenabeele wrote:

> Is there a specified order for the enumeration? (I presume not, but curious).

As the implementation uses a hash, it would be the same order, i.e. order in which they were set.

Thomas Sawyer wrote:

> If #each_pair, why not #each?

The nice thing about each_pair is that is common to both Hash and Struct and would have the same meaning for OpenStruct. Struct#each yields only the values, so that might cause some confusion?

> It was also recommended to me to suggest #to_h here.

I think it would be nice to have this, and not only in OpenStruct. Matz seems positive about it too [ruby-core:43363].

Here's what I have so far:

[https://github.com/marcandre/ruby/compare/ostruct](https://github.com/marcandre/ruby/compare/ostruct)

---

**#16 - 10/27/2012 04:42 AM - ko1 (Koichi Sasada)**

ping. status?

---

**#17 - 10/29/2012 06:35 AM - marcandre (Marc-Andre Lafortune)**

- Status changed from Assigned to Closed

This issue was solved with changeset r37372. Tomas, thank you for reporting this issue. Your contribution to Ruby is greatly appreciated.

May Ruby be with you.

---

- `lib/ostruct.rb` (each_pair): Add #each_pair [#1400]