

## Ruby - Feature #8229

### extend Hash.include?

04/07/2013 08:14 AM - eike.rb (Eike Dierks)

<b>Status:</b>	Assigned	
<b>Priority:</b>	Normal	
<b>Assignee:</b>	matz (Yukihiro Matsumoto)	
<b>Target version:</b>		
<b>Description</b>		
<p>I'd like to suggest to extend the Hash.include? method.</p> <p>Currently Hash.include? can only be used to ask for a key, I believe it should be extended to ask for a (key value) pair.</p> <p>I believe this extension can be done without breaking prior api.</p> <p>I suggest to extend the signature of Hash.include? to Hash.include?(key, value)</p> <p>That message should return true, if the receiving object does have an object at key which is equal to value.</p> <p>It would be a simple replacement for: h.include?(key) &amp;&amp; h[key] == value</p> <p>But I do not want to stop there. I'm heading for h.include_all?(other_hash) and h.include_any?(other_hash)</p> <p>and it would be valuable to have h.intersect(other_hash) etc</p> <p>I believe these to be useful primitives when working with hashes.</p> <p>I'd like to have the api of the Set class available for the Hash class as well, but there working on key/value matching.</p> <p>Obviously any change to such the substantial class as the Hash class needs a lot of thought for compatibility.</p> <p>But I believe this can be done without breaking any prior code, and it could add a lot of new out of the box functionality.</p> <p>This probably needs some more thought. We might come up with some dsl like thing like h.includes.any? or h.includes.all? or h.includes.none? to be used cross all collection classes.</p> <p>Someone must be in charge for the Hash class, my 2p</p>		

#### History

##### #1 - 04/09/2013 04:08 PM - naruse (Yui NARUSE)

- Description updated
- Category changed from misc to core
- Assignee changed from nobu (Nobuyoshi Nakada) to matz (Yukihiro Matsumoto)

##### #2 - 04/09/2013 04:37 PM - nobu (Nobuyoshi Nakada)

- File 0001-hash.c-Hash-include-improve.patch added

One ticket, one feature, please.

**#3 - 04/10/2013 01:17 AM - marcandre (Marc-Andre Lafortune)**

Could you elaborate on why this is needed and in which cases one would need this?

Moreover, I would not write `h.include?(key) && h[key] == value`. In most cases `h[key] == value` is sufficient (unless value can be nil or there's a default proc). Otherwise one can write `h.fetch(key, DIFFERENT) == value`, where DIFFERENT is an value different from all possible values of the hash, like `Object.new`).

**#4 - 04/12/2013 05:02 AM - headius (Charles Nutter)**

As a feature that affects all Ruby implementations, this should probably move to CommonRuby: <https://bugs.ruby-lang.org/projects/common-ruby>

**#5 - 07/04/2013 01:10 AM - fuadksd (Fuad Saud)**

I think this would be more interesting if in the form:

```
h = { a: 'b', c: { d: 'e' } }  
  
h.include?({c: { d: 'e' } }) # => true
```

It would accept a hash and check whether h includes that hash.

**#6 - 12/10/2020 08:53 AM - naruse (Yui NARUSE)**

- Target version deleted (3.0)

**#7 - 04/03/2024 03:50 AM - hsb (Hiroshi SHIBATA)**

- Status changed from Open to Assigned

**Files**

0001-hash.c-Hash-include-improve.patch	5.37 KB	04/09/2013	nobu (Nobuyoshi Nakada)
--	---------	------------	-------------------------