Ruby master - Feature #8499

Importing Hash#slice, Hash#slice!, Hash#except, and Hash#except! from ActiveSupport

06/06/2013 04:12 PM - mrkn (Kenta Murata)

Status:	Closed	
Priority:	Normal	
Assignee:	matz (Yukihiro Matsumoto)	
Target version:	2.5	

Description

According to my experiences, the following two idioms often appeare in application codes.

1. hash = other_hash.select { |k, | [:key1, :key2, :key3].include? k }
2. hash = other_hash.reject { |k, | [:key1, :key2, :key3].include? k }

On Rails, they can be written in the following forms by using ActiveSupport's features.

1. hash = other_hash.slice(:key1, :key2, :key3)
2. hash = other_hash.except(:key1, :key2, :key3)

I think the latter forms are shorter and more readable than the former ones.

So I propose to import the following methods from ActiveSupport:

- Hash#slice
- Hash#slice!
- Hash#except
- Hash#except!

Related issues:

Related to Ruby master - Feature #9108: Hash sub-selections	**Closed**
Related to Ruby master - Feature #12461: Hash & keys to make subset.	**Rejected**
Related to Ruby master - Feature #13563: Implement Hash#choice method.	**Closed**
Related to Ruby master - Feature #15863: Add `Hash#slice!` and `ENV.slice!`	**Rejected**
Has duplicate Ruby master - Feature #15822: Add Hash#except	**Closed**

Associated revisions

Revision 6c50bdda - 10/21/2017 06:08 AM - glass

hash.c: Add Hash#slice

- hash.c (rb_hash_slice): add Hash#slice [Feature #8499]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@60229 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 60229 - 10/21/2017 06:08 AM - Glass_saga (Masaki Matsushita)

hash.c: Add Hash#slice

- hash.c (rb_hash_slice): add Hash#slice [Feature #8499]

Revision 60229 - 10/21/2017 06:08 AM - glass

hash.c: Add Hash#slice

- hash.c (rb_hash_slice): add Hash#slice [Feature #8499]

Revision 60229 - 10/21/2017 06:08 AM - glass

hash.c: Add Hash#slice

- hash.c (rb_hash_slice): add Hash#slice [Feature #8499]

History

#1 - 06/06/2013 04:13 PM - sorah (Sorah Fukumori)

+1

#2 - 06/06/2013 04:23 PM - zzak (Zachary Scott)

Hello,

On Thu, Jun 6, 2013 at 4:12 PM, mrkn (Kenta Murata) muraken@gmail.com wrote:

> On Rails, they can be written in the following forms by using ActiveSupport's features.
>
> 1. hash = other_hash.slice(:key1, :key2, :key3)
> 2. hash = other_hash.reject(:key1, :key2, :key3}

Do you mean "other_hash.except(...)"? There is already Hash#reject

#3 - 06/06/2013 04:27 PM - nobu (Nobuyoshi Nakada)

- Description updated

#4 - 06/06/2013 04:28 PM - mrkn (Kenta Murata)

zzak (Zachary Scott) wrote:

> Hello,
>
> On Thu, Jun 6, 2013 at 4:12 PM, mrkn (Kenta Murata) muraken@gmail.com wrote:
>
> > On Rails, they can be written in the following forms by using ActiveSupport's features.
> >
> > 1. hash = other_hash.slice(:key1, :key2, :key3)
> > 2. hash = other_hash.reject(:key1, :key2, :key3}
>
> Do you mean "other_hash.except(...)"? There is already Hash#reject

Yes, It's my mistake!

How can I edit the issue description?

#5 - 06/06/2013 04:51 PM - mrkn (Kenta Murata)

- Description updated

I could fix the description.

#6 - 06/06/2013 04:59 PM - nobu (Nobuyoshi Nakada)

Or enhance the existing methods?

#7 - 06/06/2013 05:10 PM - mrkn (Kenta Murata)

nobu (Nobuyoshi Nakada) wrote:

> Or enhance the existing methods?

I think Hash#[] with the multiple arguments can be an alternative of Hash#slice.

#8 - 06/06/2013 05:20 PM - charliesome (Charlie Somerville)

+1

#9 - 06/06/2013 10:43 PM - nobu (Nobuyoshi Nakada)

mrkn (Kenta Murata) wrote:

> I think Hash#[] with the multiple arguments can be an alternative of Hash#slice.

Hash#[] should return the values, not the pairs.

#10 - 06/06/2013 10:45 PM - rosenfeld (Rodrigo Rosenfeld Rosas)

+1

#11 - 06/07/2013 12:15 AM - prijutme4ty (Ilya Vorontsov)

mrkn (Kenta Murata) wrote:

> nobu (Nobuyoshi Nakada) wrote:
>
> > Or enhance the existing methods?
>
> I think Hash#[] with the multiple arguments can be an alternative of Hash#slice.

There was a proposal (can't find it) to make indexing by multiple arguments to work as nested hashes so that hsh[:a,:b,:c] works as (hsh[:a] && hsh[:a][:b] && hsh[:a][:b][:c]). Your proposal automatically blocks this proposal.

#12 - 06/07/2013 12:38 AM - mrkn (Kenta Murata)

prijutme4ty (Ilya Vorontsov) wrote:

> mrkn (Kenta Murata) wrote:
>
> > nobu (Nobuyoshi Nakada) wrote:
> >
> > > Or enhance the existing methods?
> >
> > I think Hash#[] with the multiple arguments can be an alternative of Hash#slice.
>
> There was a proposal (can't find it) to make indexing by multiple arguments to work as nested hashes so that hsh[:a,:b,:c] works as (hsh[:a] && hsh[:a][:b] && hsh[:a][:b][:c]). Your proposal automatically blocks this proposal.

My proposal primaliry consists of Hash#slice, Hash#except, and bang versions of them.
Hash#[] is optional.

#13 - 06/07/2013 12:53 AM - claytrump (Clay Trump)

Yay, +1 for slice & except

BTW, makes no sense to me if h[:foo, :bar] returns keys and values in a
hash while h[:foo] returns a value.

<lay trum/>

#14 - 06/07/2013 03:22 AM - vipulnsward (Vipul Amler)

+1

#15 - 06/07/2013 07:53 AM - mrkn (Kenta Murata)

On Fri, Jun 7, 2013 at 1:50 AM, Zachary Scott zachary@zacharyscott.net wrote:

> Please update description if the proposal has changed (ie: Hash#[])

I mentioned Hash#[] as the reply to nobu's comment #6.
It has not changed my proposal.
My proposal only consists of slice, slice!, except, and except!.

--
Kenta Murata
OpenPGP FP = 1D69 ADDE 081C 9CC2 2E54 98C1 CEFE 8AFB 6081 B062

#16 - 06/07/2013 10:37 AM - nobu (Nobuyoshi Nakada)

I meant Hash#select and Hash#reject by "the existing methods".

i.e.:

```
hash = other_hash.select(:key1, :key2, :key3)
```

```
hash = other_hash.reject(:key1, :key2, :key3)
```

But Hash#slice and Hash#except seems fine.

#17 - 07/26/2013 10:58 PM - Glass_saga (Masaki Matsushita)

- File patch.diff added

How about this implementation?

#18 - 07/27/2013 03:22 PM - matz (Yukihiro Matsumoto)

The slice method (Array#slice) retrieve "a slice of elements" from an Array.
Considering that, slice is *not* a good name for the behavior.

So, I prefer Nobu's idea in comment #16

```
hash = other_hash.select(:key1, :key2, :key3)
hash = other_hash.reject(:key1, :key2, :key3)
```

Matz

#19 - 07/27/2013 03:25 PM - znz (Kazuhiro NISHIYAMA)

In attached patch.diff

```
assert_equal({1=>2, 3=>4}, h.slice!(1, 3))
```

but ActiveSupport's h.slice!(1, 3) returns {5=>6}.

http://api.rubyonrails.org/classes/Hash.html#method-i-slice-21

#20 - 07/28/2013 05:56 AM - nobu (Nobuyoshi Nakada)

I've missed the returned values until I've implemented it actually.

- In ActiveSupport
  - Hash#slice! keeps the given keys and returns removed key/value pairs.
  - Hash#except! removes the given keys and returns self.
- In this proposal
  - Hash#slice! removes the given keys and returns removed key/value pairs.
  - Hash#except! is same as ActiveSupport.
- Existing methods
  - Hash#select! keeps the given (by the block) keys and returns self or nil.
  - Hash#reject! removes the given (by the block) keys and returns self or nil.

I don't think changing the result semantics by if any arguments are given is good idea.
What I've thoutht about Hash#slice! was Hash#extract! in ActiveSupport actually.
So what about Hash#extract, Hash#except and those !-versions?

#21 - 07/28/2013 06:26 PM - mrkn (Kenta Murata)

The attached file is not a part of my proposal. It made by Glass_saga. My proposal is the same as ActiveSupport.

#22 - 07/29/2013 12:24 PM - Glass_saga (Masaki Matsushita)

- File patch2.diff added

I'm sorry for my wrong implementation.
patch2.diff makes Hash#slice! and #except! behave the same as ActiveSupport.
However, I think it isn't good idea to import Hash#slice! and #except! from ActiveSupport as it is.

Because:

- They returns self if no changes were made. It is inconsistent with other built-in methods like #select! and #reject!.
- #slice! returns rest of hash, not slice of hash like following. It may be confusing.

```
hash = {1=>2,3=>4,5=>6}
hash.slice!(1,3) #=> {5,6}
hash #=> {1=>2,3=>4}
```

#23 - 01/30/2014 06:17 AM - hsbt (Hiroshi SHIBATA)

- Target version changed from 2.1.0 to 2.2.0

#24 - 06/22/2014 03:14 PM - nobu (Nobuyoshi Nakada)

- Description updated

Another name, Hash#only.
http://blog.s21g.com/articles/228

#25 - 06/12/2015 07:32 AM - ko1 (Koichi Sasada)

- Related to Feature #9108: Hash sub-selections added

#26 - 11/07/2015 08:17 PM - keithrbennett (Keith Bennett)

I was about to report this feature request but was happy to find that it was already there...but disappointed that it's been here so long. I'm using Ruby but not Rails and have been needing this functionality a lot lately.

Unless I'm misunderstanding, the implementation of such methods would be trivial. I would much rather see this implemented with a nonoptimal name than not implemented at all.

'slice' and 'except' are ok with me, but if there is a concern about their correctness as names, perhaps they could be 'select_if_key_in' and 'reject_if_key_in'.

But again, anything would be better than nothing.

#27 - 07/19/2016 07:46 AM - shyouhei (Shyouhei Urabe)

- Related to Feature #12461: Hash & keys to make subset. added

#28 - 08/31/2017 06:12 AM - mrkn (Kenta Murata)

- Related to Feature #13563: Implement Hash#choice method. added

#29 - 09/07/2017 10:26 AM - Glass_saga (Masaki Matsushita)

- Target version set to 2.5

- File patch3.diff added

I just rebased it to trunk.

#30 - 09/19/2017 05:42 AM - Glass_saga (Masaki Matsushita)

- File patch4.diff added

Improved implementation and test.
Added documentation.

#31 - 10/19/2017 07:28 AM - knu (Akinori MUSHA)

One concern is that Array#slice! returns deleted elements whereas this Hash#slice! would return pairs left after slicing.

#32 - 10/19/2017 07:48 AM - mrkn (Kenta Murata)

- Status changed from Assigned to Closed

As Hash#slice has been accepted in #13563, I decided to close this issue.
I intend to open the different issue for discussing other methods such as Hash#except.

Thanks.

#33 - 05/05/2019 03:37 AM - nobu (Nobuyoshi Nakada)

- Has duplicate Feature #15822: Add Hash#except added

#34 - 05/21/2019 03:35 AM - nobu (Nobuyoshi Nakada)

- Related to Feature #15863: Add `Hash#slice!` and `ENV.slice!` added

Files

patch.diff	4.05 KB	07/26/2013	Glass_saga (Masaki Matsushita)
patch2.diff	4.07 KB	07/29/2013	Glass_saga (Masaki Matsushita)
patch3.diff	4.04 KB	09/07/2017	Glass_saga (Masaki Matsushita)
patch4.diff	4.5 KB	09/19/2017	Glass_saga (Masaki Matsushita)