Add to_proc on Hash

Procs can be called the same way a hash is with []. But a Hash is not mappable as a Proc.

```ruby
my_hash = ->key{
  a: 1, b: 2, c: 3, d: 4, e: 5, f: 6
}[key]

my_hash[:a]
# => 1

[e, , a, :b, :f, :c, :d].map(&my_hash)  # hash is now mappable
# => [5, 1, 2, 6, 3, 4]
```

This seems so straight forward I believe it should be part of the language itself with the .to_proc method call.

Associated revisions

Revision fbe967ec - 11/10/2015 07:57 AM - nobu (Nobuyoshi Nakada)
hash.c: to_proc
- hash.c (rb_hash_to_proc): new method Hash#to_proc. [Feature #11653]

Revision 52520 - 11/10/2015 07:57 AM - nobu (Nobuyoshi Nakada)
hash.c: to_proc
- hash.c (rb_hash_to_proc): new method Hash#to_proc. [Feature #11653]

Revision 52520 - 11/10/2015 07:57 AM - nobu (Nobuyoshi Nakada)
hash.c: to_proc
- hash.c (rb_hash_to_proc): new method Hash#to_proc. [Feature #11653]

Revision 52520 - 11/10/2015 07:57 AM - nobu (Nobuyoshi Nakada)
hash.c: to_proc
- hash.c (rb_hash_to_proc): new method Hash#to_proc. [Feature #11653]

Revision 52520 - 11/10/2015 07:57 AM - nobu (Nobuyoshi Nakada)
hash.c: to_proc
- hash.c (rb_hash_to_proc): new method Hash#to_proc. [Feature #11653]

Revision a56b0f8b - 11/10/2015 09:22 AM - nobu (Nobuyoshi Nakada)
bm_hash_to_proc.rb
- benchmark/bm_hash_to_proc.rb: benchmark for Hash#to_proc. [Feature #11653]
Revision 52522 - 11/10/2015 09:22 AM - nobu (Nobuyoshi Nakada)

bm_hash_to_proc.rb

- benchmark/bm_hash_to_proc.rb: benchmark for Hash#to_proc. [Feature #11653]

Revision 52522 - 11/10/2015 09:22 AM - nobu (Nobuyoshi Nakada)

bm_hash_to_proc.rb

- benchmark/bm_hash_to_proc.rb: benchmark for Hash#to_proc. [Feature #11653]

Revision 52522 - 11/10/2015 09:22 AM - nobu (Nobuyoshi Nakada)

bm_hash_to_proc.rb

- benchmark/bm_hash_to_proc.rb: benchmark for Hash#to_proc. [Feature #11653]

Revision 52522 - 11/10/2015 09:22 AM - nobu (Nobuyoshi Nakada)

bm_hash_to_proc.rb

- benchmark/bm_hash_to_proc.rb: benchmark for Hash#to_proc. [Feature #11653]

Revision 52522 - 11/10/2015 09:22 AM - nobu (Nobuyoshi Nakada)

bm_hash_to_proc.rb

- benchmark/bm_hash_to_proc.rb: benchmark for Hash#to_proc. [Feature #11653]

Revision 16e7869d - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]

Revision 52524 - 11/10/2015 09:25 AM - nobu (Nobuyoshi Nakada)

hash.c: use rb_func_proc_new

- hash.c (rb_hash_to_proc): use rb_func_proc_new to make light weight proc. [Feature #11653]
I think this is an excellent idea. I hope this can make it into Ruby 2.3.

#2 - 11/04/2015 06:01 AM - nobu (Nobuyoshi Nakada)
- Description updated

You can write it as:

```
[:e, :a, :b, :f, :c, :d].map(&my_hash.method(:[]))
```

#3 - 11/09/2015 08:03 AM - matz (Yukihiro Matsumoto)
- Assignee changed from matz (Yukihiro Matsumoto) to nobu (Nobuyoshi Nakada)

Accepted.

Matz.

#4 - 11/09/2015 08:08 AM - ko1 (Koichi Sasada)

Discussion: [https://docs.google.com/document/d/1D0Eo5N7NE-unlySOKG9iVI_eVFI66BQPM4PKp7NvMyQ/pub](https://docs.google.com/document/d/1D0Eo5N7NE-unlySOKG9iVI_eVFI66BQPM4PKp7NvMyQ/pub)

Feel free to continue discussion on this ticket.

#5 - 11/10/2015 07:57 AM - nobu (Nobuyoshi Nakada)
- Status changed from Open to Closed

Applied in changeset r52520.

```ruby
hash.c: to_proc
  * hash.c (rb_hash_to_proc): new method Hash#to_proc. [Feature #11653]
```

#6 - 06/24/2016 08:34 AM - jwmittag (Jörg W Mittag)

Daniel P. Clark wrote:

```
my_hash = ->key{|
  a: 1, b: 2, c: 3, d: 4, e: 5, f: 6
}[
key]

my_hash[:a]
# => 1

[:e, :a, :b, :f, :c, :d].map(&my_hash) # hash is now mappable
# => [5, 1, 2, 6, 3, 4]
```

This seems so straight forward I believe it should be part of the language itself with the .to_proc method call.

This is basically a subset of what I proposed a year ago in #11262. I additionally proposed that Hash also implement call. IMO, it doesn't make much sense to have one without the other: both methods basically say "hey, I'm kinda like a function", and a Hash is basically just a function from keys to elements. I proposed the same thing for Array and Set, which are essentially also just functions from indices to elements (Array) or elements to booleans (Set). However, so far, there has been no interest in that ticket.

#7 - 06/25/2016 05:01 AM - duerst (Martin Dürst)

Jörg W Mittag wrote:

```
This is basically a subset of what I proposed a year ago in #11262.
```

Commenting on a closed issue won't help much to move an open issue forward. One thing that might help is to split Feature #11262 into smaller issues that can be discussed more easily.

A Hash is basically just a function from keys to elements.

Yes. The most direct case is Hash.new { [h, k] my_function(k) }. :-)

05/10/2020