Ruby master - Misc #16436

hash missing #last method, make it not so consistent (it has #first)

12/19/2019 04:21 PM - zw963 (Wei Zheng)

Status:	Open
Priority:	Normal
Assignee:	

Description

```
(pry):main(0)> h = {x:1, y:2, z:3}
{
    :x => 1,
    :y => 2,
    :z => 3
}
(pry):main(0)> h.first
[
    :x,
    1
]
(pry):main(0)> h.first.first
:x
(pry):main(0)> h.first.last
1
```

last method not exist.

```
(pry):main(0)> h.last

Exception: NoMethodError: undefined method `last' for {:x=>1, :y=>2, :z=>3}:Hash
```

We have to use #to_a to make last working.

```
(pry):main(0)> h.to_a.last
[
    :z,
    3
]

(pry):main(0)> h.to_a.last.first
:z
(pry):main(0)> h.to_a.last.last
3
```

History

#1 - 12/19/2019 04:27 PM - zverok (Victor Shepelev)

Hash doesn't have #first either :)

```
Hash.instance_method(:first)
# => #<UnboundMethod: Hash(Enumerable)#first(*)>
#                          ^^^^^^^^^^
```

The first method existence is just because of hash being a Enumerable, not because "first pair(s) of hash" seen as having some frequently used meaning.

(As a side note, Ruby probably could've had BidirectionalEnumerable for collections which can be enumerated both directions without side-effects; this one would be a good place to have #last method.)

#2 - 12/19/2019 04:37 PM - sawa (Tsuyoshi Sawada)

What is this issue? Is it just a note to everyone, or are you claiming this to be a bug, or is it a feature request?

#3 - 12/20/2019 12:37 AM - shevegen (Robert A. Heiler)

sawa wrote:

> What is this issue? Is it just a note to everyone, or are you claiming this to be a bug, or is it a feature request?

I think it could only possibly be a feature request since it is not a bug. But you are uite right that it is difficult to infer from the issue description alone.

To the topic itself: I can see some points, in particular with hashes being ordered, where you could use .first and .last similar as how could be done to Array. I once wondered whether we should also have .second, .third and so forth, but one problem I had with that is that this becomes less and less likely to use; for example, if you have a large array of array, how often would you use .eight (or [7]) for that matter? Probably not that often. I do however have found .second and .third in use now and then.

Of course we can use [] instead anyway, but the code layout was sometimes strange. Such as in:

```
first_element  = array[0]
last_element   = array[-1]
second_element = array[1]
```

versus

```
first_element  = array.first
last_element   = array.last
second_element = array[1] # <- this one looks a bit weird if used with .first and .last
```

However had, I think I have not created an issue request. I assume there may have been reasons why we do not have .second or similar by default.

Note that the name BidirectionalEnumerable is quite weird too. This is how we may end up with things such as HashWithIndifferentAccess. That's an even stranger name ... ;)

Sorry for digressing. May have to see when zw963 can reply and clarify.

#4 - 12/20/2019 03:18 AM - zw963 (Wei Zheng)

sawa (Tsuyoshi Sawada) wrote:

> What is this issue? Is it just a note to everyone, or are you claiming this to be a bug, or is it a feature request?

Sorry, this issue not a bug, i knew Hash#first come from Enumerable, the reason i issue it is: i love ruby language, use it days and days, when encountering some inconsistencies things, i expect to give some feedback, and what i thought.

So, for this issue, the question is:

why Array include Enumerable module, and implement Array#first Array#last method, if we should do same things for Hash?
make this language more elegant?

#5 - 12/29/2019 03:44 AM - zw963 (Wei Zheng)

- Subject changed from hash missing #last method, make it not so consistent (it has first) to hash missing #last method, make it not so consistent (it has #first)