Proposal: Shorthand operator for "#instance_method"

10/23/2019 12:47 AM - osyo (manga osyo)

Status: Open
Priority: Normal
Assignee: 
Target version: 

Description
hi, created issues to discuss shorthand for "#instance_method"

Overview
Ruby 2.7 adds a #method shorthand .: operator.

- Feature #12125: Proposal: Shorthand operator for Object#method - Ruby master - Ruby Issue Tracking System
- Feature #13581: Syntax sugar for method reference - CommonRuby - Ruby Issue Tracking System

In this issues want to discuss the shorthand operator for "#instance_method".

Background
If you want to pass an array to Array#zip or Hash#merge as shown below, the code will be messy.

arrays = [['a', 'b'], ['c'], ['d', 'e']
hashs = [{'a' => 1, 'b' => 2, 'c' => 3}, {'d' => 4, 'e' => 5}]

# It's long.
arrays.first.zip(*arrays.drop(1))
arrays.first.product(*arrays.drop(1))
hashs.first.merge(*hashs.drop(1))

But #instance_method is long. I'm thinking shorthand operator for #instance_method.

:MEMO: There was a suggestion to add Array.zip or Array.product in the past

- Feature #8970: Array.zip and Array.product - Ruby master - Ruby Issue Tracking System
- Feature #6499: Array::zip - Ruby master - Ruby Issue Tracking System
- Feature #7444: Array#product_set - Ruby master - Ruby Issue Tracking System

Proposal new operator
I can't come up with a suitable operator yet, sorry...
However, considering the :: operator, I think "operator + Symbol(:)" is good.

# Document-like writing
# `#` + operator
Array#zip
Array.#zip
Array#:zip
# Constant-link writing
#
# :: + :hoge
Array:::.zip
Array::@zip

Also, the following syntax is valid at this time, so it may be difficult to adopt.

Array!zip
Array@zip
Array&zip
Array:zip

Please comment if you are interested in the shorthand operator for # instance_method.

- Other #instance_method usecase
- Proposal shorthand operator
- etc...

Thank you! :)

---

**Associated revisions**

Revision 5aa0e6be - 01/13/2020 12:22 PM - znz (Kazuhiro NISHIYAMA)
Mention new feature of Hash#transform_keys [Feature #16273]
ref b25e27277dc39f25cfa4db8452d254f6cc8046e

**History**

#1 - 10/23/2019 06:45 AM - nobu (Nobuyoshi Nakada)
I don't think using # acceptable, and have thought :::

#2 - 10/23/2019 06:49 AM - naruse (Yui NARUSE)
I don't think Array.instance_method(zip).bind_call("arrays) should be recommended, and worth adding shorthand operator.

#3 - 10/23/2019 07:12 AM - mrkn (Kenta Murata)
I don't think the first examples are messy.
On the contrary, the second examples using bind_call are quite messy than the first.

#4 - 10/23/2019 07:50 AM - zverok (Victor Shepelev)
Funnily enough (and not completely intentionally) the problem is "solved" with #16264:

```ruby
::zip.call(*arrays)
# or even... for those who likes to cry "code golf!"
::zip.(*arrays)
```

Currently, you also can

```ruby
:zip.to_proc.call(*arrays)
```

Though, I tend to agree with mrkn (Kenta Murata) that "no-tricks" first example looks pretty clear... But I understand that in codebase where this metaphor emerges regularly, it could become tiresome and too non-atomic to read.

One another non-orthodox suggestion (which uses #then loathed by some, but of all the rest of examples the only one which reads directly "zip first array with the rest of them"):

```ruby
arrays.then { |first, *rest| first.zip(*rest) }
```

#5 - 01/13/2020 12:23 PM - znz (Kazuhiro NISHIYAMA)

- Status changed from Open to Closed

Applied in changeset git|5aa0e6bee916f454ecf886252e1b025d824f7bd8.

---

Mention new feature of Hash#transform_keys [Feature #16273]
ref b25e27277dc39f25cfa4db8452d254f6cc8046e
- Status changed from Closed to Open