Ruby trunk - Feature #9453

Return symbols of defined methods for `attr` and friends

01/26/2014 05:55 PM - jballanc (Joshua Ballanco)

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

Description

With Ruby 2.1 returning a symbol from def and define_method, that leaves attr, attr_reader, attr_writer, and attr_accessor as ways to define methods that still return nil. This is unfortunate, because it prevents the use of method decorators developed to work with def from also working with the attr* methods. Because these mechanisms can define more than one method, the return values would need to be arrays of symbols.

For an example of how this could be useful in real-world code, consider this sample from James Edward Gray II's Warehouse Keeper example (https://github.com/JEG2/warehouse_keeper):

```
attr_reader :images, :key_map, :window, :screen_manager, :animations
private     :images, :key_map, :window, :screen_manager, :animations
```

if attr_reader returned symbols, then this could be simplified to:

```
private *attr_reader(:images, :key_map, :window, :screen_manager, :animations)
```

I've attached a patch that implements this change and includes a few tests. For those who use git, I've also submitted this as a pull request here: https://github.com/ruby/ruby/pull/517

Related issues:

Has duplicate Ruby trunk - Feature #13560: Module#attr_ methods return reason... **Open**

History

#1 - 02/03/2014 03:05 PM - jballanc (Joshua Ballanco)

bump

Would be nice if we could get this in 2.2 (if someone could update the target version, I'd appreciate it).

#2 - 02/03/2014 03:13 PM - Eregon (Benoit Daloze)

```
private *attr_reader(:images, :key_map, :window, :screen_manager, :animations)
```

does not look so nice to me (the parens and the explicit splat).
Maybe #private should accept an Array of Symbols so:

```
private attr_reader :images, :key_map, :window, :screen_manager, :animations
```

If there are so much ":", it might also be worth using %i:

```
private attr_reader %i[images key_map window screen_manager animations]
```

but then attr_reader would also need to accept an Array of Symbols.

#3 - 02/03/2014 06:44 PM - sawa (Tsuyoshi Sawada)

What is the point of defining a private accessor method? You can directly refer to the instance variables without using accessors.

#4 - 02/04/2014 05:22 AM - avdi (Avdi Grimm)

There are a number of reasons I use them, but the most obvious is that a
misspelled accessor method will raise NoMethod error, whereas a misspelled
@ivar reference will silently return nil.

--
Avdi Grimm
http://avdi.org

I only check email twice a day. to reach me sooner, go to http://awayfind.com/avdi

#5 - 02/05/2014 12:27 AM - jballanc (Joshua Ballanco)

Tsuyoshi Sawada wrote:

> What is the point of defining a private accessor method? You can directly refer to the instance variables without using accessors.

The example I gave was just one case of "code in the wild" that would benefit from this change. Undoubtedly, as more people begin to take advantage of the ability to build method decorators (now that method definitions return something other than nil), having the attr* family of methods return something meaningful will mean that they can also benefit from these decorators.

#6 - 03/27/2014 05:13 PM - jballanc (Joshua Ballanco)

It's been almost 2 months...any chance we could get a comment on this from the core team?

#7 - 03/28/2014 03:55 PM - usa (Usaku NAKAMURA)

- Status changed from Open to Assigned

- Assignee set to matz (Yukihiro Matsumoto)

#8 - 03/29/2014 02:58 PM - matz (Yukihiro Matsumoto)

- Status changed from Assigned to Rejected

I am not positive. The example

private *attr_reader(:images, :key_map, :window, :screen_manager,:animations)

is not really intuitive. Besides that, you can define private_attr_reader, etc. by yourself.
In that case, the code will be

private_attr_reader :images, :key_map, :window, :screen_manager,:animations

and looks much better.

Matz

#9 - 04/11/2014 07:48 PM - trans (Thomas Sawyer)

Letting #private accept an Array seems more preferable then adding yet another ~~method~~ slew of methods: private_attr_writer, private_attr_accessor, protected_attr_reader, protected_attr_writer, protected_attr_accessor, ...

#10 - 04/12/2014 01:31 AM - matz (Yukihiro Matsumoto)

Thomas, private method that accept an array for methods names would be an interesting idea worth discussion.
But it is different issue.

Matz.

#11 - 08/08/2014 10:38 PM - nerdrew (Andrew Lazarus)

Chaining the two suggestions leads to a nice syntax (in my opinion):

private attr_reader :name, :address, :etc

private needs to accept an array and attr_* needs to return its list (as an array) of arguments.

#12 - 03/11/2017 05:43 AM - vais (Vais Salikhov)

attr_* methods returning nil should be considered a bug at this point, since all other ways of defining methods return a symbol. This makes Ruby inconsistent, and violates its own Principle of Least Surprise.

#13 - 04/17/2017 08:29 AM - matz (Yukihiro Matsumoto)

Yes, def and define_method returns symbols now.
But it does not mean attr_* should return symbols. Since they can define multiple methods.
Considering there's no use for private attributes, I don't think the proposal creates real-world value.

Besides that, mention to the principle of least surprise gives you negative evaluation here (since the background varies from user to user, and it does not lead to constructive discussion).

Matz.

#14 - 05/14/2017 04:50 AM - shyouhei (Shyouhei Urabe)

- Has duplicate Feature #13560: Module#attr_ methods return reasonable values added

Files

attr_rv.patch	3.23 KB	01/26/2014	jballanc (Joshua Ballanco)