Ruby master - Feature #10183

An alternative name for method `class`

08/29/2014 12:52 PM - sawa (Tsuyoshi Sawada)

Status:	Open
Priority:	Normal
Assignee:	
Target version:	

Description

The method class is special in that it always has to have an explicit receiver in order to avoid crash with the keyword class. But this is very inconvenient. I have seen so many

```
self.class
```

in codes. I propose that there should be an alternative name for this method so that it can be used with an implicit receiver, and the method name class should be gradually depricated.

As for the name, I have no clear idea. I can only think of klass, but perhaps someone else might come up with a better name.

History

#1 - 08/29/2014 01:40 PM - trans (Thomas Sawyer)

I once suggested object_class to go along with object_id.

Also, see https://bugs.ruby-lang.org/issues/6478 which talks about BasicObject#__class__.

#2 - 08/29/2014 02:02 PM - sawa (Tsuyoshi Sawada)

Thomas Sawyer wrote:

> I once suggested object_class to go along with object_id.
>
> Also, see https://bugs.ruby-lang.org/issues/6478 which talks about BasicObject#__class__.

What is __class__?

#3 - 08/29/2014 06:33 PM - shevegen (Robert A. Heiler)

The problem with object_class is that it is longer than self.class

Remember that object_id used to be called id in the past before that was renamed.

#4 - 10/30/2017 03:56 AM - wardrop (Tom Wardrop)

Came here to make a similar feature suggestion. Ideally class would be aliased as something such as klass which has become somewhat of a ruby idiom to avoid collision with the class keyword. Though I'd also be interested to discuss the possibility of introducing shorter syntax for self., such as some kind of single-character prefix. Some examples:

```
self.class::MYCONST

klass::MYCONST # alias of #class method

$class::MYCONST # special reserved global variable

~class::MYCONST # tilda prefix shouldn't clash with any existing syntax rules
```

#5 - 10/31/2017 11:33 AM - duerst (Martin Dürst)

My guess is that self.class mostly appears in the context of metaprogramming. Metaprogramming is already not as concise as plain programs anyway, so the self.class may not be that much of an issue. Using class in general code should be avoided anyway, and replaced with resopnd_to?,...

Also, with something like klass, all Ruby users would have to learn this special case.

#6 - 10/31/2017 11:38 AM - zverok (Victor Shepelev)

My guess is that self.class mostly appears in the context of metaprogramming.

Why? In simple inheritance, you may have cases like:

```
self.class::CONST # the constant is redefined in subclasses

def clone_with_tricks(...)
  self.class.new(...) # produce exactly the same class as self
end
```

...and a lot of others.

#7 - 10/31/2017 11:42 AM - zverok (Victor Shepelev)

BTW, I'd rather have some special syntax for generic case "want to use local name, but it is keyword". For example (syntax is bad, but shows usage):

```
def my_method(if:, retry: false)
  if %if.call # `if` variable from local context, not `if` keyword
    ....

  %class # locally-available `class` method, not `class` keyword

  return false unless %retry
end
```