### Keyword `object` to be used instead of `class <<`

03/07/2013 02:59 PM - alexeymuranov (Alexey Muranov)

<table>
<thead>
<tr>
<th>Status</th>
<th>Rejected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee</td>
<td></td>
</tr>
<tr>
<td>Target version</td>
<td>3.0</td>
</tr>
</tbody>
</table>

#### Description

```ruby
I propose to introduce a new keyword (({object})) and to use it instead of the mysterious (({class <<})): 

```object` foo
```def bar
    # ...
end
def baz
    # ...
end
eend
```

instead of

```ruby
foo = Object.new
class << foo
```def bar
    # ...
end
def baz
    # ...
end
eend
```

#### History

**#1 - 03/07/2013 03:23 PM - david_macmahon (David MacMahon)**

As an alternative, how about allowing Object#extend to take a block that would be executed with the context such that methods defined therein would become singleton methods of the object receiving #extend?

Your example would then become:

```ruby
foo = Object.new
foo.extend do
  def bar
    # ...
  end
  def baz
    # ...
  end
end
```

Sorry if this is impossible for some reason I don't know of,

Dave

On Mar 6, 2013, at 9:59 PM, alexeymuranov (Alexey Muranov) wrote:

> Issue [#8038](https://bugs.ruby-lang.org/issues/8038) has been reported by alexeymuranov (Alexey Muranov).
I propose to introduce a new keyword `((object))` and to use it instead of the mysterious `((class <<))`:

```ruby
object foo
  def bar
    # ...
  end
  def baz
    # ...
  end
end
```

instead of

```ruby
foo = Object.new
class << foo
  def bar
    # ...
  end
  def baz
    # ...
  end
end
```

http://bugs.ruby-lang.org/

---

#2 - 03/07/2013 03:26 PM - matz (Yukihiro Matsumoto)

- Status changed from Open to Rejected

Introducing a new keyword would break a lot of existing programs. We don't want to do unless there's strong requirement. Making Object#extend to take a block is interesting idea. We will discuss later if it's submitted.

Matz.

#3 - 03/07/2013 05:01 PM - trans (Thomas Sawyer)

@alexey I sympathize. I tend to avoid class << foo too b/c it "reads" so poorly.

david (david he) Facets extends #extend to do that actually, but it does not add the the methods directly. Instead it creates an anonymous module which it includes into the singleton class. In this way it behaves the same as extending with a module.

#4 - 03/08/2013 11:52 AM - phluid61 (Matthew Kerwin)

matz (Yukihiro Matsumoto) wrote:

Introducing a new keyword would break a lot of existing programs. We don't want to do unless there's strong requirement. Making Object#extend to take a block is interesting idea. We will discuss later if it's submitted.

Matz.

I'm interested in this discussion, including Facets' interpretation. I just created #8046