Ruby - Feature #6672

Calling #() without dot before braces

06/30/2012 07:13 AM - prijutme4ty (Ilya Vorontsov)

Status:	Rejected
Priority:	Normal
Assignee:	
Target version:	

Description

=begin
It looks odd to call Proc/Method (({pr})) using (({pr.(*args)})) or (({pr[*args]})) syntax. Why not to use (({pr(*args)})) syntax? In such a case methods(procs) would become nearer to first-class object and calls are more standardized such that there's much less difference between method call and call of (({#call})) method

So I suggest syntax (({x.(*args,&block)})) alias to (({x(*args,&block)}))
It will make it possible to transparently redefine any method in a scope like this:
class String
def my_meth(direction)
index = method :rindex if direction == 'RtoL'

... lots of code that uses #index method

```
end

def my_second_meth
  upcase = method :downcase  # we decided to try what if we change all upcases to downcases in thi
s method and added such a line
```

lots of code using upcase() method

```
  upcase()  # it's sad that it's impossible not to use brackets at all, but this'd be ambiguous
end
```

end

Also I wrote about syntax like (({:meth.(*args)})) which creates a proxy-object having to_proc method - so that it's possible to write (({[1,2,3].map &:to_s.(2)})) -- now no dot in such syntax

Also it'd be possible to implement some object with syntax like (({obj(args1)(args2)})) - obj has method (({#call})) that returns object that also has method (({#call})) and they are both called.
It can be in such a way: (({method(:index)('z')}))

One problem is that it can behave differently from current behavior in case that method have the same name as a local-variable. Now it works in such a way:
p='var';
print p #=> 'var' ## works as local variable
p('hi') #=> 'hi'   ## works as method call

I don't know if it's a spec, I suppose that one mustn't use both variable and method at the same place. So even it's a spec it can be revised in future versions of ruby so that new behaviour would be like that:
p='var';
print p #=> 'var' ## works as local variable
p('hi') #=> undefined method call for 'var':String
=end

Related issues:

Is duplicate of Ruby - Feature #7346: object(...) as syntax sugar for object....	**Rejected**	**11/13/2012**

History

#1 - 06/30/2012 08:04 AM - nobu (Nobuyoshi Nakada)

- Description updated

FYI, this is a feature which had been implemented once and reverted in the past.

> I don't know if it's a spec, I suppose that one mustn't use both variable and method at the same place. So even it's a spec it can be revised in future versions of ruby so that new behaviour would be like that:

Yes, it's a spec, and there was so many code depending this, more than expected before the try.

#2 - 06/30/2012 09:10 AM - prijutme4ty (Ilya Vorontsov)

nobu (Nobuyoshi Nakada) wrote:

> FYI, this is a feature which had been implemented once and reverted in the past.
>
> > I don't know if it's a spec, I suppose that one mustn't use both variable and method at the same place. So even it's a spec it can be revised in future versions of ruby so that new behaviour would be like that:
>
> Yes, it's a spec, and there was so many code depending this, more than expected before the try.

Sadly. It looks that code basing on such a spec isn't written in a good manner =\

#3 - 11/18/2012 12:37 PM - Anonymous

In that case this issue should be closed. (I also think this feature is too much sugar.)

#4 - 11/19/2012 12:26 AM - matz (Yukihiro Matsumoto)

- Status changed from Open to Rejected