Ruby master - Feature #7346

**object(...) as syntax sugar for object.call(...)**

11/13/2012 08:50 PM - rosenfeld (Rodrigo Rosenfeld Rosas)

<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>matz (Yukihiro Matsumoto)</td>
</tr>
<tr>
<td>Target version:</td>
<td>3.0</td>
</tr>
</tbody>
</table>

**Description**

I propose for the parser to interpret "object(...)" as "object.call(...)". It should raise NoMethodError at runtime if object doesn't respond to "call".

This would read better than using "call":

```ruby
double = -> n { n * 2 }
double(3) == 6
```

Sorry if this has already been proposed before (and rejected) but I couldn't find any references to something like this using Redmine's search interface.

**Related issues:**

Has duplicate Ruby master - Feature #6672: Calling #() without dot before braces - Rejected - 06/30/2012

**History**

**#1 - 11/13/2012 08:59 PM - matz (Yukihiro Matsumoto)**

- Status changed from Open to Rejected

I have once tried, but it caused serious incompatibility problem for example:

```ruby
p = Object.new
p(15)
```

So compromise with object.() syntax introduced in 1.9.

Matz.

**#2 - 11/13/2012 09:02 PM - rosenfeld (Rodrigo Rosenfeld Rosas)**

Ah, ok, I didn't know about this syntax until now. What does the code above do?

**#3 - 11/13/2012 09:15 PM - matz (Yukihiro Matsumoto)**

We easily forget conflict between method names and variable names, in a language like Ruby, where methods and variables have separated name space.

We expect p(15) to print 15 even when we have a variable named p in the scope.

Matz.

**#4 - 11/13/2012 10:02 PM - rosenfeld (Rodrigo Rosenfeld Rosas)**

Ah, of course! :D I totally forgot about Kernel#p! :P

Yes, that makes total sense.

**#5 - 11/14/2012 04:23 AM - nathan.f77 (Nathan Broadbent)**

rosenfeld (Rodrigo Rosenfeld Rosas), I'll just mention that you can use Proc#[] in your example:

```ruby
double = -> n { n * 2 }
double[3] == 6 #=> true
```

On Wednesday, 14 November 2012, rosenfeld (Rodrigo Rosenfeld Rosas) wrote:

02/21/2020
Ah, of course! :D I totally forgot about Kernel#p! :P

Yes, that makes total sense.

Feature #7346: object(...) as syntax sugar for object.call(...)
https://bugs.ruby-lang.org/issues/7346#change-32860

Author: rosenfeld (Rodrigo Rosenfeld Rosas)
Status: Rejected
Priority: Normal
Assignee: matz (Yukihiro Matsumoto)
Category: core
Target version: Next Major

I propose for the parser to interpret "object(...)" as "object.call(...)".
It should raise NoMethodError at runtime if object doesn't respond to "call".

This would read better than using "call":

double = -> n { n * 2 }
double(3) == 6

Sorry if this has already been proposed before (and rejected) but I couldn't find any references to something like this using Redmine's search interface.

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

#6 - 11/14/2012 04:57 AM - rosenfeld (Rodrigo Rosenfeld Rosas)
Yes, I know, it is just that I prefer to read object.call(arguments) than object[arguments]. This is just a personal opinion, I know and I can change my mind some day about this :}