Subject: Re: Bug or desired behavior in lambda functions?
Posted by penteado on Mon, 08 Jun 2015 20:31:43 GMT
View Forum Message <> Reply to Message

The best workaround I found is using call_function:

```
IDL> expr='x: x^2'
IDL> l=call_function('lambda',expr)
IDL> l(3)
      9
```

On Monday, June 8, 2015 at 2:34:12 PM UTC-3, Paulo Penteado wrote:

> Hello,
>
> Despite what the documentation might suggest, and one might expect, it seems there is no way to use string variables to make lambda functions.
>
> If I try to make a lambda function with the code inside lambda(), all is well:
>
> ```
> IDL> l=lambda('x: x^2')
> IDL> l(2)
>            4
> ```
>
> However, if I put the same code in a string and pass it to lambda, it does not accept the code:
>
> ```
> IDL> expr='x: x^2'
> IDL> l=lambda(expr)
> % LAMBDA: Code must be of the form "arg1,arg2,... : statement"
> ```
>
> I expect this is caused by IDL parser's special behavior when it encounters lambda(), which is what allows this
>
> ```
> IDL> l=lambda(x: x^2)
> ```
>
> So I think when I put a variable name inside the lambda() call, the parser thinks is a string literal, and tries to parse the variable name as code.
>
> Is this intended behavior, or a bug?
>
> Paulo