Subject: 'True' python call
Posted by Helder Marchetto on Fri, 04 Sep 2015 13:31:22 GMT
View Forum Message <> Reply to Message

Hi,
[disclaimer1: I'm quite python ignorant...]

I'm trying this call to a function where I have to make a call using the "True" keyword, so that the call should look like this:
res = obj.method(keyword=True)

However, I cannot do this in IDL because it will tell me that the variable true is not defined. Very well. So I've tried using the following synthax options:
res = obj.method(keyword=1b)
res = obj.method(keyword="True")
res = obj.method(keyword='True')
res = obj.method(/keyword)
res = obj.method(keyword=Python.Run("True"))

All fail. Any suggestion how to do this?

The best try was this one:
[disclaimer2: save all your data. This will make the IDL session (IDLDE or command line) crash]

IDL> Python("True")
% Loaded DLM: PYTHON27.
% Type conversion error: Unable to convert given STRING to Long64.
% Detected at: PYTHON::INIT 19 C:\Program
Files\Exelis\IDL85\lib\bridges\python__define.pro

In fact any string will cause the same error/crash.
The reason is to be found in line 19 of python__define.pro (as shown above) where the parameter passed to the initialization of Python is only checked if it exists, not if it is a number.

Cheers,
Helder

Subject: Re: 'True' python call
Posted by Fabzi on Fri, 04 Sep 2015 14:29:44 GMT
View Forum Message <> Reply to Message

On 09/04/2015 03:31 PM, Helder wrote:
> All fail. Any suggestion how to do this?

I dont have IDL8.5 to test it but in python "1" is equivalent to "True"

Subject: Re: 'True' python call
Posted by Helder Marchetto on Fri, 04 Sep 2015 14:47:19 GMT
View Forum Message <> Reply to Message

On Friday, September 4, 2015 at 4:29:48 PM UTC+2, Fabien wrote:
> On 09/04/2015 03:31 PM, Helder wrote:
>> All fail. Any suggestion how to do this?
>
> I dont have IDL8.5 to test it but in python "1" is equivalent to "True"

Hi Fabien,
in the meanwhile, I'm realizing that this may be a very different error. The error I get in IDL is:

% PYTHON_CALLMETHOD: Exception: 'crosscheck' is an invalid keyword argument for this function.

I'm using BFMatcher of opencv (ver 3.0) and the keyword crosscheck is defined as crossCheck with capital "C" and the error it throws is of an invalid keyword...

I just found a dirty workaround:

Python.Run, 'import cv2'
Python.Run, 'bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)'
bf = Python.bf

and it seems to work.

Still, the question holds: can one call methods with keywords with capital letters?

Cheers,
Helder

Subject: Re: 'True' python call
Posted by chris_torrence@NOSPAM on Tue, 08 Sep 2015 16:07:49 GMT
View Forum Message <> Reply to Message

On Friday, September 4, 2015 at 8:47:21 AM UTC-6, Helder wrote:
> On Friday, September 4, 2015 at 4:29:48 PM UTC+2, Fabien wrote:
>> On 09/04/2015 03:31 PM, Helder wrote:
>>> All fail. Any suggestion how to do this?
>>
>> I dont have IDL8.5 to test it but in python "1" is equivalent to "True"
>
> Hi Fabien,
> in the meanwhile, I'm realizing that this may be a very different error. The error I get in IDL is:
>
> % PYTHON_CALLMETHOD: Exception: 'crosscheck' is an invalid keyword argument for this

function.
>
> I'm using BFMatcher of opencv (ver 3.0) and the keyword crosscheck is defined as crossCheck with capital "C" and the error it throws is of an invalid keyword...
>
> I just found a dirty workaround:
>
> Python.Run, 'import cv2'
> Python.Run, 'bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)'
> bf = Python.bf
>
> and it seems to work.
>
> Still, the question holds: can one call methods with keywords with capital letters?
>
> Cheers,
> Helder

Hi Helder,

The short answer is "no". Right now, IDL converts all Python keywords to lowercase. However, if you know the order of the arguments, you can call the method without using the keyword names. For example:

```
IDL> Python.Run, "def hello(x, mykey1=4, MYKEY2=5):\n" + $
IDL>   "  return 'x='+str(x)+'  mykey1=' + str(mykey1) + '  MYKEY2=' + str(MYKEY2)"

IDL> print, Python.hello(1,mykey1=2,MYKEY2=3)  ; this will throw an error

IDL> print, Python.hello(1,2,3)  ; this works!
```

Or, like you discovered, you can call the method as a string. One tip - you can use ">>>" to avoid typing "Python.run"

```
IDL> >>>hello(1, mykey1=2, MYKEY2=3)
'x=1  mykey1=2  MYKEY2=3'
```

Note: When calling functions and methods, the IDL Python bridge attempts to find the correct case - lower, upper, mixed. So that works fine. However, for Python keywords, it's much trickier to get the function's "signature". I found a stackoverflow here:
http://stackoverflow.com/questions/2677185/how-can-i-read-a-functions-signature-including-default-argument-values

Maybe this is something we can try for the next release.

Cheers,
Chris