Python编程提高：如何调用DLL函数之传递数值、指针与字符串参数

编码那些事

发布时间：18-10-25 14:33

> 在Python语言中，可以使用ctypes模块调用其它如C++语言编写的动态链接库DLL文件中的函数，在提高软件运行效率的同时，也可以充分利用目前市面上各种第三方的DLL库函数，以扩充Python软件的功能及应用领域，减少重复编写代码、重复造轮子的工作量，这也充分体现了Python语言作为一种胶水语言所特有的优势。

这次以具体的例子讲一下在Python中，如何使用ctypes模块调用DLL中的库函数。本文的编程系统环境是win7 64位，Python使用的版本是python2.7.14。

由于DLL中函数中传递的参数类型比较多样化，拟打算分三次讲解这部分内容，这次先讲传递数值、指针与字符串参数的情况，后面再分两次讲解传递结构体、数值数组等类型的情况。

DLL文件的加载

假定已经有了一个DLL文件“MyDll.dll”，其函数约定的调用方式为C调用（cdecl）方式，则Python中加载该dll文件的代码如下：

其中，第1行是引入ctypes模块，第2行是采用C调用约定加载“MyDll.dll”文件，并将返回值赋给dll变量，后续调用该DLL文件中的函数时，会使用该变量定义要使用的具体函数。

另外，需要说明的是，若DLL函数的调用约定是标准调用约定（stdcall）方式，则DLL文件的加载代码改为如下：

dll = WinDLL('MyDll.dll')

DLL函数的调用——函数参数为数值情况

如对于“MyDll.dll”文件中的函数add，其函数声明如下：

```
#define EXPORTFUC extern "C" __declspec(dllexport)
EXPORTFUC int add(int x,int y);
```


该函数有两个int类型的输入参数数x和y，返回的两个数的和。其C语言的实现代码如下：

```
3  int add(int x, int y)
4  {
5      return x + y;
6  }
```

在Python中的调用方式如下：

```
16  # int add(int x,int y)
17  z = dll.add(1, 7)
18  print(z)
```


这个函数应该说是最简单的一个函数了，在第17行，直接使用第一步加载DLL后返回的名称dll，后面跟函数名字即可返回其值。

DLL函数的调用——函数参数为指针情况

对于上面的函数改进为add2，其函数C语言的实现代码如下：

```
8   void add2(int *x, int *y, int *z)
9   {
10      *z = *x + *y;
11  }
```


此时函数有三个指向int类型的指针参数x、y、z，z为x和y的和。

在Python中的调用方式如下：

```
20  x = c_int(2)
21  y = c_int(3)
22  z = c_int(0)
23  dll.add2(byref(x), byref(y), byref(z))
24  print(z.value)
```


其中，第20-22行定义了3个int型的变量x、y和z，初始值分别为2，3，0。第23行调用add2函数时，使用byref指明参数传递时为引用传递，对应着C语言的指针传递。函数运行后，使用z.value即可查看z的值。

也可以使用下面的代码调用：

```
20  x = c_int(2)
21  y = c_int(3)
22  z = c_int(0)
23  add2 = dll.add2
24  add2.argtypes = [POINTER(c_int), POINTER(c_int), POINTER(c_int)]
25  add2(x,y,z)
26  print(z.value)
```

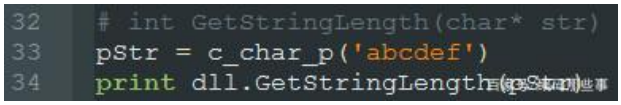

上面代码中，第23-24行，在使用add2函数时，先将函数赋给一个变量add2，然后对其输入输出参数进行单独声明，使用POINTER声明为这三个参数为指向int类型的指针变量。

DLL函数的调用——函数参数为字符串情况

例1：如对于下面的函数，返回一个输入字符串的字节长度，其函数C语言的实现代码如下：

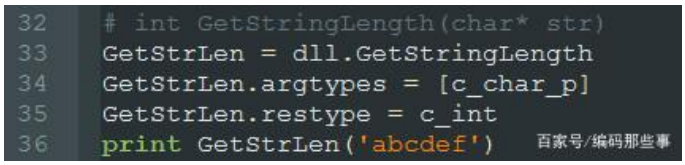
```
13  int GetStringLength(char* str)
14  {
15      string tmp(str);
16      return tmp.length();
17  }
```


在Python中的调用代码如下：



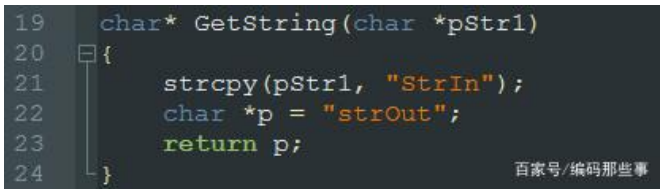
其中，第33行使用c_char_p定义了一个指向char型的指针变量pStr，并赋初值为'abcdef'，第34行将其传入GetStringLength函数返回其长度。

也可以使用下面代码调用：

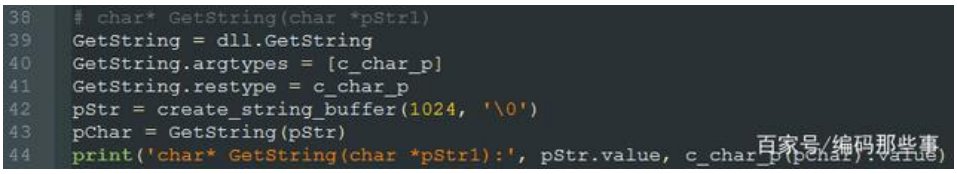


将GetStringLength函数的输入输出参数分别使用argtypes和restype单独进行声明。

例2：如对于下面的函数，输入输出皆为字符串指针，函数的功能是对于输入pStr1赋值为“StrIn”，对于输出返回一个指向字符串常量“strOut”的指针，其函数C语言的实现代码如下：



在Python中的调用代码如下：



在上面代码中，同样分别对输入输出参数进行了声明。对于输入参数pStr，使用create_string_buffer函数定义了一个字符串缓冲区。对于返回值pChar，在打印输出结果时，将其强制转换为c_char_p类型，取其value值即可。

完整的测试代码

完整的测试代码如下图所示：

```python
# -*- coding: utf-8 -*-#

#-------------------------------------------------------------------------------
# Name:         使用cyptes模块调用dll
# Description:  Python使用cyptes模块调用dll函数之传递数值/指针/字符串
# Author:       lgk
# Date:         2018/10/25
#-------------------------------------------------------------------------------

from __future__ import print_function
from ctypes import *
import numpy as np
from struct import *

dll = CDLL('MyDll.dll')

# int add(int x,int y)
z = dll.add(1, 7)
print('int add(int x,int y):', z)

# void add2(int *x, int *y, int *z)
x = c_int(2)
y = c_int(3)
z = c_int(0)
dll.add2(byref(x), byref(y), byref(z))
print('void add2(int *x, int *y, int *z):', z.value)

add2 = dll.add2
add2.argtypes = [POINTER(c_int), POINTER(c_int), POINTER(c_int)]
add2(x,y,z)
print('void add2(int *x, int *y, int *z):', z.value)

# int GetStringLength(char* str)
str1 = 'abcdef'
length = dll.GetStringLength(str1)
print('int GetStringLength(char* str):', length)

# char* GetString(char *pStr1)
GetString = dll.GetString
GetString.argtypes = [c_char_p]
GetString.restype = c_char_p
pStr = create_string_buffer(1024, '\0')
pChar = GetString(pStr)
print('char* GetString(char *pStr1):', pStr.value, c_char_p(pChar).value)
```


运行结果如下图所示：

```
Python解释器
*** Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:25:58) [MSC v.1500 64 bit (AMD64)] on win32. ***
*** Remote Python引擎  处于激活状态 ***
>>>
*** 远程解释器再初始化了  ***
>>>
int add(int x,int y): 8
void add2(int *x, int *y, int *z): 5
void add2(int *x, int *y, int *z): 5
int GetStringLength(char* str): 6
char* GetString(char *pStr1): StrIn strOut
>>>
```


总结

这次的例子基本涵盖了在Python中通过ctypes模块调用DLL函数时，传递数值、指针、字符串类型参数时的大部分情况。要注意的是，使用ctypes映射C语言中的数据类型时，两者必须完全一致。下面是Python的ctypes模块中数据类型与C语言中数据类型对照表：

`ctypes` defines a number of primitive C compatible data types:

ctypes type	C type	Python type
c_bool	_Bool	bool (1)
c_char	char	1-character string
c_wchar	wchar_t	1-character unicode string
c_byte	char	int/long
c_ubyte	unsigned char	int/long
c_short	short	int/long
c_ushort	unsigned short	int/long
c_int	int	int/long
c_uint	unsigned int	int/long
c_long	long	int/long
c_ulong	unsigned long	int/long
c_longlong	__int64 or long long	int/long
c_ulonglong	unsigned __int64 or unsigned long long	int/long
c_float	float	float
c_double	double	float
c_longdouble	long double	float
c_char_p	char * (NUL terminated)	string or None
c_wchar_p	wchar_t * (NUL terminated)	unicode or None
c_void_p	void *	int/long or None