



链滴

# JAVA | JAVA 混编 C++，JNA 采坑记录

作者: [fpdan](#)

原文链接: <https://ld246.com/article/1592613400633>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

# 我的环境

JDK 1.8.0\_241 64位

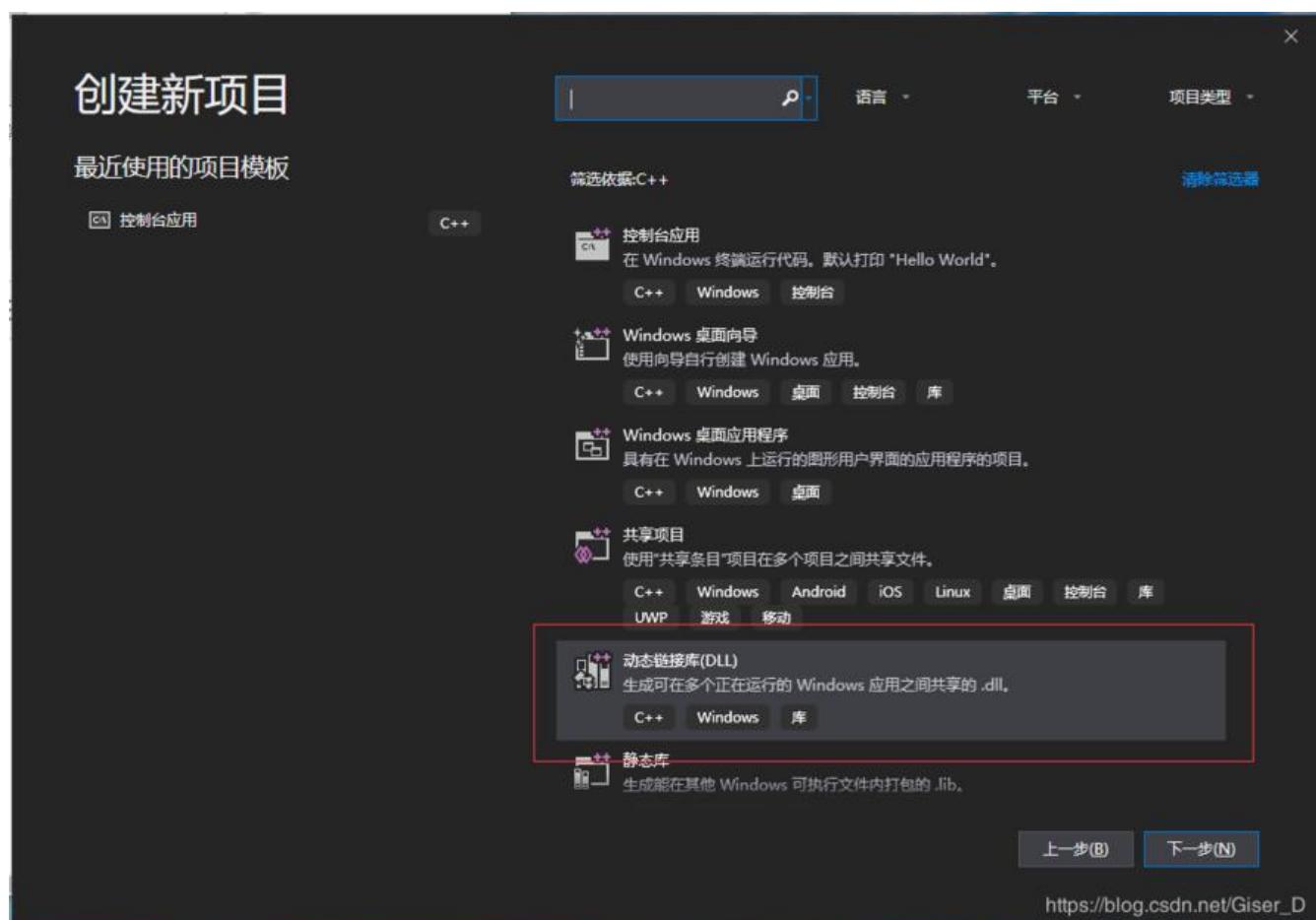
```
C:\Users\Administrator\Desktop\jna-func-master>java -version  
java version "1.8.0_241"  
Java(TM) SE Runtime Environment (build 1.8.0_241-b07)  
Java HotSpot(TM) 64-Bit Server VM (build 25.241-b07, mixed mode)
```

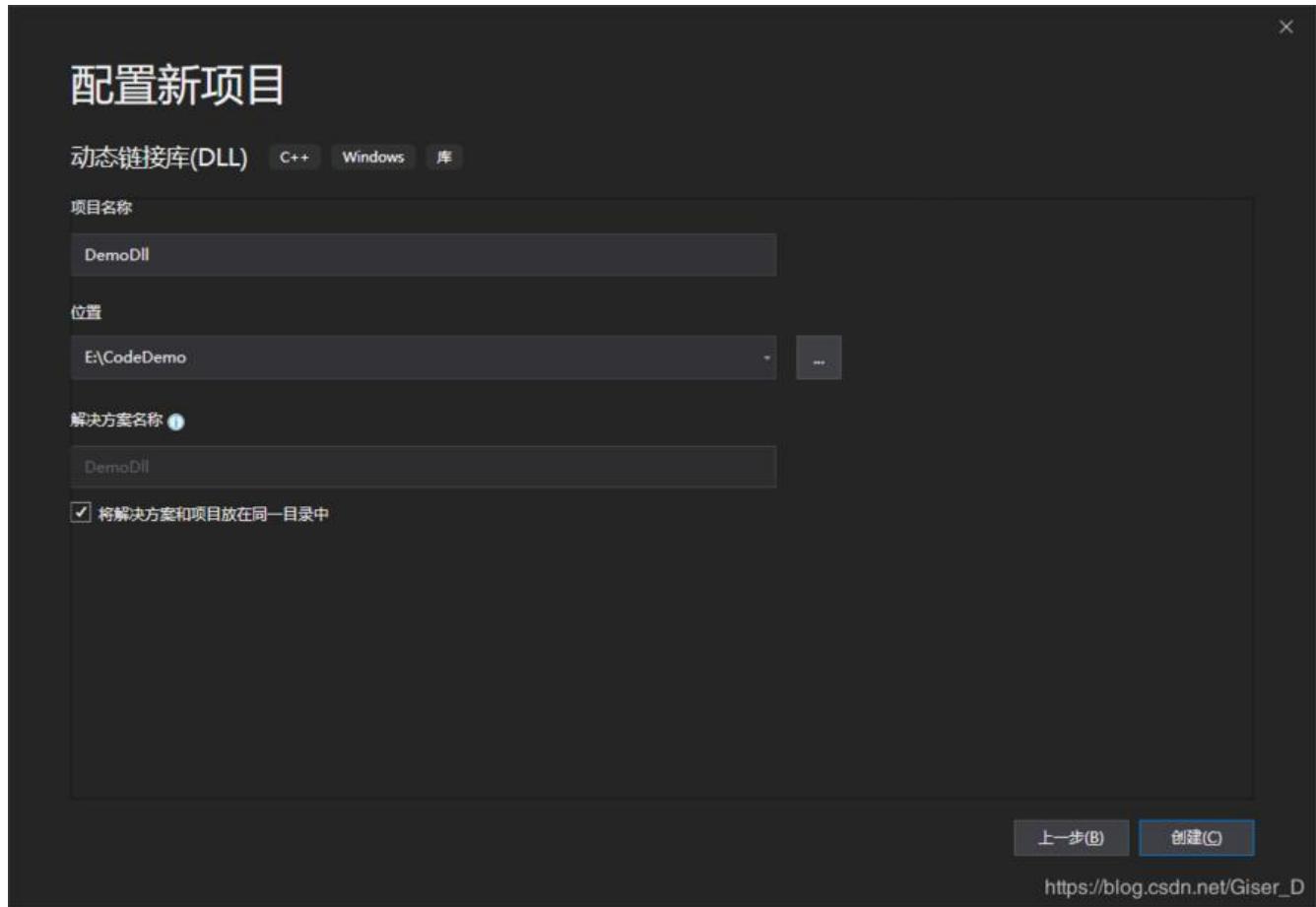
VS 2019

Windows 10

大体思路是，C++ 生成 dll 动态链接库，Java 调用 dll。

## C++





抄一个 c++ 代码

参考：[https://blog.csdn.net/Giser\\_D/article/details/89677441](https://blog.csdn.net/Giser_D/article/details/89677441)

pch.h

```
// pch.h: 这是预编译标头文件。  
// 下方列出的文件仅编译一次，提高了将来生成的生成性能。  
// 这还将影响 IntelliSense 性能，包括代码完成和许多代码浏览功能。  
// 但是，如果此处列出的文件中的任何一个在生成之间有更新，它们全部都将被重新编译。  
// 请勿在此处添加要频繁更新的文件，这将使得性能优势无效。  
  
#ifndef PCH_H  
#define PCH_H  
  
// 添加要在此处预编译的标头  
#include "framework.h"  
  
#endif //PCH_H  
  
// 定义宏  
#ifdef IMPORT_DLL  
#else  
#define IMPORT_DLL extern "C" _declspec(dllexport) // 指的是允许将其给外部调用  
#endif  
  
IMPORT_DLL int add(int a, int b);  
IMPORT_DLL int minus(int a, int b);
```

```
IMPORT_DLL int multiply(int a, int b);
IMPORT_DLL double divide(int a, int b);
```

pch.cpp

// pch.cpp: 与预编译标头对应的源文件

```
#include "pch.h"
```

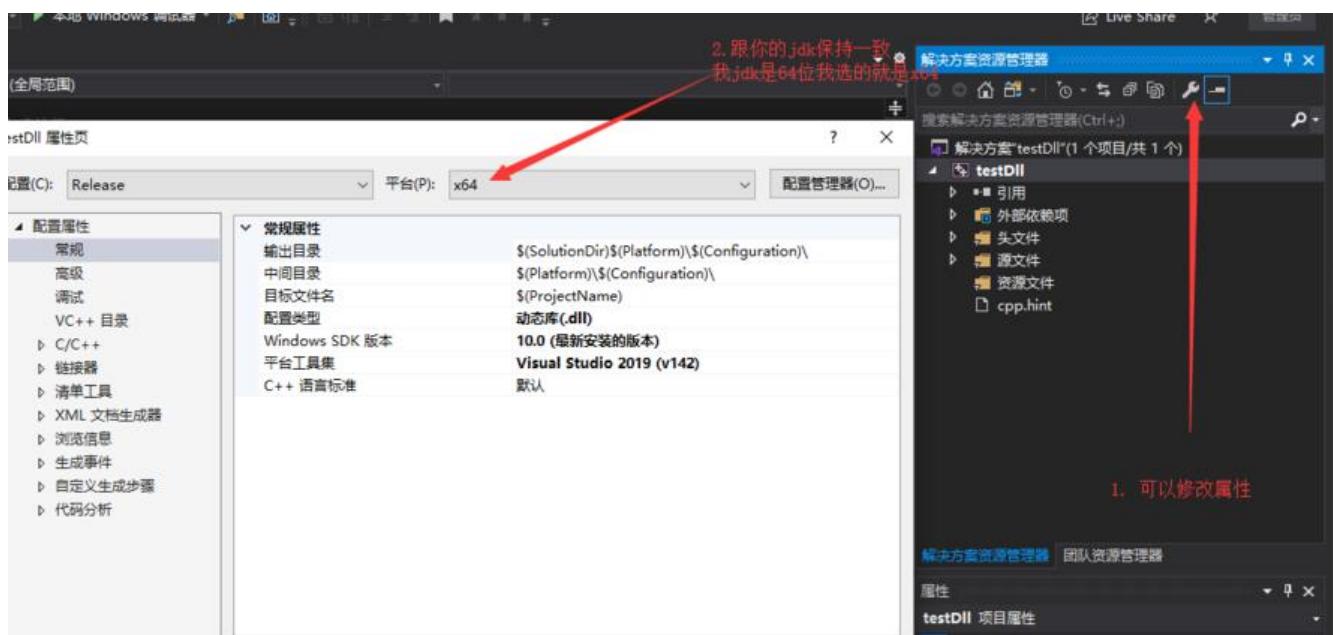
// 当使用预编译的头时，需要使用此源文件，编译才能成功。

```
int add(int a, int b)
{
    return a + b;
}
```

```
int minus(int a, int b)
{
    return a - b;
}
```

```
int multiply(int a, int b)
{
    return a * b;
}
```

```
double divide(int a, int b)
{
    double m = (double)a / b;
    return m;
}
```



右击项目生成

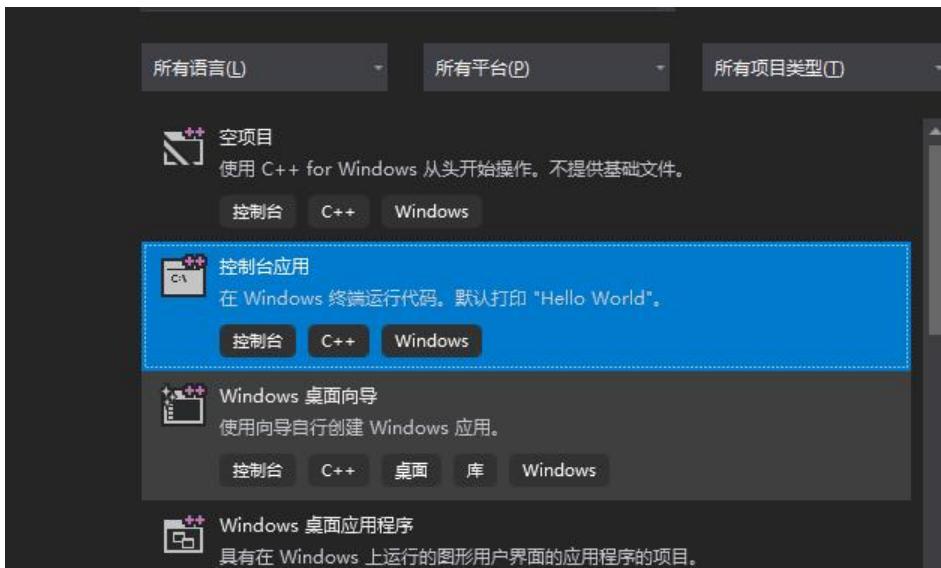
```

输出
显示输出来源(S): 生成
1>D:\Users\Administrator\Desktop\testDll\pch.cpp(8,1): warning C4273: "add": dll 链接不一致
1>D:\Users\Administrator\Desktop\testDll\pch.h(23,16): message : 参见 "add" 的前一个定义
1>D:\Users\Administrator\Desktop\testDll\pch.cpp(13,1): warning C4273: "minus": dll 链接不一致
1>D:\Users\Administrator\Desktop\testDll\pch.h(24,16): message : 参见 "minus" 的前一个定义
1>D:\Users\Administrator\Desktop\testDll\pch.cpp(18,1): warning C4273: "multiply": dll 链接不一致
1>D:\Users\Administrator\Desktop\testDll\pch.h(25,16): message : 参见 "multiply" 的前一个定义
1>D:\Users\Administrator\Desktop\testDll\pch.cpp(23,1): warning C4273: "divide": dll 链接不一致
1>D:\Users\Administrator\Desktop\testDll\pch.h(26,19): message : 参见 "divide" 的前一个定义
1>dllmain.cpp
1>正在生成库 D:\Users\Administrator\Desktop\testDll\x64\Debug\testDll.lib 和对象 D:\Users\Administrator\Desktop\testDll\x64\Debug\testDll.exp
1>testDll.vcxproj -> D:\Users\Administrator\Desktop\testDll\x64\Debug\testDll.dll
1>已完成生成项目 "testDll.vcxproj" 的操作。
全部重新生成: 成功 1 个, 失败 0 个, 跳过 0 个

```

控制台可以看到你 dll 生成的目录，生成完测试一下。

## 1. 生成一个控制台程序



### ConsoleApplication1.cpp

```

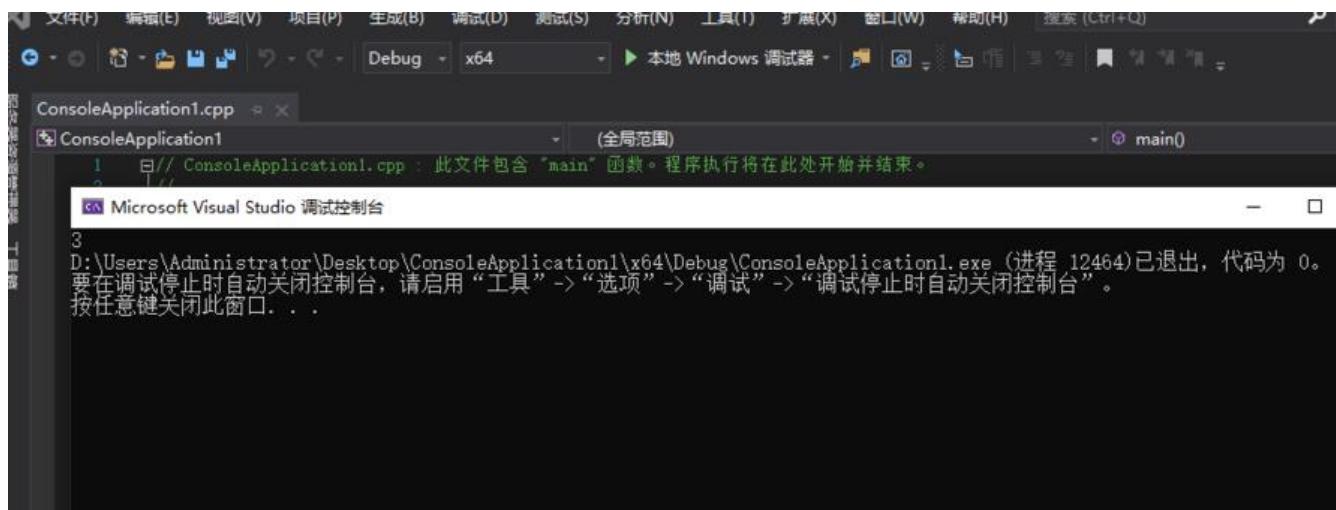
// ConsoleApplication1.cpp : 此文件包含 "main" 函数。程序执行将在此处开始并结束。
//

#include <iostream>
#include <Windows.h>

int main()
{
    HINSTANCE hDllInst;
    hDllInst = LoadLibrary(L"C:\\\\Users\\\\Administrator\\\\Desktop\\\\testDll.dll"); //调用DLL
    typedef int(*PLUSFUNC)(int a, int b);
    PLUSFUNC plus_str = (PLUSFUNC)GetProcAddress(hDllInst, "add"); //GetProcAddress为获
    该函数的地址
    std::cout << plus_str(1, 2);
    // std::cout << "Hello World!\\n";
}

run trollface

```



## Java

JNA 调用 dll

导包

```
<dependency>
    <groupId>net.java.dev.jna</groupId>
    <artifactId>jna</artifactId>
    <version>5.3.1</version>
</dependency>

import com.sun.jna.Library;
import com.sun.jna.Native;

public class TestDll {
    public interface MyLibrary extends Library {
        // testDll 为 dll 名称
        Hello.MyLibrary INSTANCE = Native.load("testDll", Hello.MyLibrary.class);
        int add(int a, int b);
        int minus(int a, int b);
        int multiply(int a, int b);
        double divide(int a, int b);
    }

    public static void main(String[] args) {
        int test = Hello.MyLibrary.INSTANCE.add(1,1);
        System.out.println(test);
    }
}
```

刚才生成的 dll 放到 jdk 目录里去

剪贴板      组织      新建      打开      选择

此电脑 > Windows10 (C:) > Program Files > Java > jdk1.8.0\_241 > bin

名称	修改日期	类型	大小
jvisualvm.exe	2020/6/19 22:28	应用程序	194 KB
keytool.exe	2020/6/19 22:28	应用程序	17 KB
kinit.exe	2020/6/19 22:28	应用程序	17 KB
klist.exe	2020/6/19 22:28	应用程序	17 KB
ktab.exe	2020/6/19 22:28	应用程序	17 KB
msvcr100.dll	2020/6/19 22:28	应用程序扩展	810 KB
native2ascii.exe	2020/6/19 22:28	应用程序	17 KB
orbd.exe	2020/6/19 22:28	应用程序	17 KB
pack200.exe	2020/6/19 22:28	应用程序	17 KB
policytool.exe	2020/6/19 22:28	应用程序	17 KB
rmi.exe	2020/6/19 22:28	应用程序	17 KB
rmid.exe	2020/6/19 22:28	应用程序	17 KB
rmiregistry.exe	2020/6/19 22:28	应用程序	17 KB
schemagen.exe	2020/6/19 22:28	应用程序	17 KB
serialver.exe	2020/6/19 22:28	应用程序	17 KB
servertool.exe	2020/6/19 22:28	应用程序	17 KB
testDll.dll	2020/6/20 7:39	应用程序扩展	57 KB
tnameserv.exe	2020/6/19 22:28	应用程序	17 KB
unpack200.exe	2020/6/19 22:28	应用程序	107 KB

右击 run trollface

```

public class TestDll {
    public interface MyLibrary extends Library {
        // DLL 为 dll 名称
        Hello.MyLibrary INSTANCE = Native.load("testDll", Hello.MyLibrary.class);
        int add(int a, int b);
        int minus(int a, int b);
        int multiply(int a, int b);
        double divide(int a, int b);
    }

    public static void main(String[] args) {
        int test = Hello.MyLibrary.INSTANCE.add(a: 1, b: 1);
        System.out.println(test);
    }
}

```

C:\Program Files\Java\jdk1.8.0\_241\bin\java.exe" ...  
 Connected to the target VM, address: '127.0.0.1:58030', transport: 'socket'  
 Disconnected from the target VM, address: '127.0.0.1:58030', transport: 'socket'  
 2  
 Process finished with exit code 0

## 坑

java.lang.UnsatisfiedLinkError: 找不到指定的模块。

生成的 dll 一定要用vs自己跑一下看看dll 生成的有没有问题，dll 要放到 jdk bin 目录下，jdk 环境变量配好。

1%不是有效的 32 位程序

看看 dll 和 jdk 是不是都是 32 或者 64

参考资料：

<https://www.jianshu.com/p/ead89497c403>

[https://blog.csdn.net/Giser\\_D/article/details/89677441](https://blog.csdn.net/Giser_D/article/details/89677441)