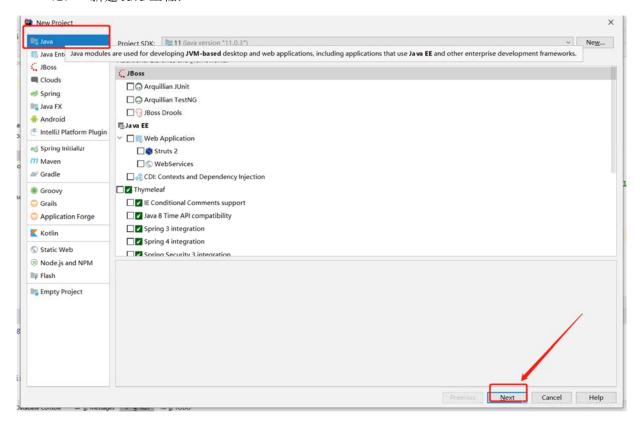
实验 4 应用 JNI 实现 Java 调用 C/C++代码编程指导

1. 创建 Java 工程

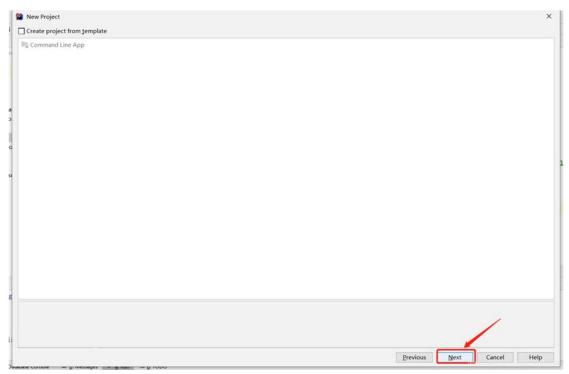
1.1 新建 Java 工程

使用 IDEA 新建 Java 演示工程,过程如下:

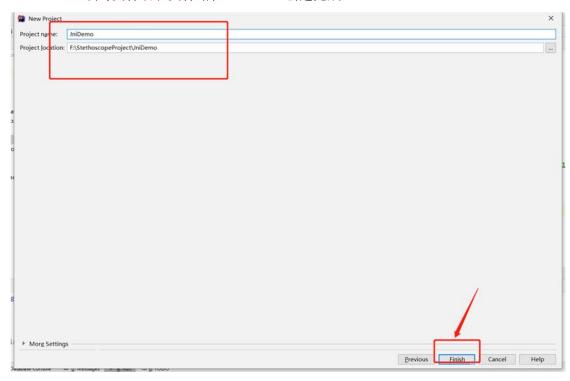
(1) 新建 Java 工程:



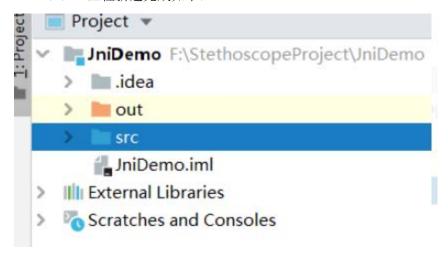
(2) 直接跳过, next:



(3) 填写文件名和文件路径,Finish,创建完成。



(4) 工程新建完成如下:



1.2 编写对应 C/C++程序的 Java 对象类

(1) 在 src 文件夹下创建 jni 包,新建类 Demo.class,内容如下:

```
IniDemo F:\StethoscopeProject\JniDemo
                                                   * @author Jianf
> idea
                                                   * @create 2020/5/12
  out
                                                   * @since 1.0.0
                                         19
   v 🛅 jni
      © Demo
                                         21
                                                  public class Main {
       C Main
  > mative
   JniDemo.iml
                                         23
                                                      public static void main(String[] args){
III External Libraries
                                         24
Scratches and Consoles
                                         26
                                                           System. load ( filename: "F: \\StethoscopePro
                                                           Demo demo = new Demo();
                                         29
                                                           demo.sayHello(x: 2, y: 3);
```

```
package jni;

public class Demo {

public native void sayHello(int x,int y);

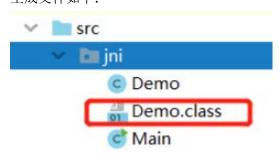
public native void sayHello(int x,int y);
```

(2) 在终端 Terminal 中生成 class 类文件:

先进入到当前 demo.java 类所在文件夹下,使用以下命令生成 class 文件:

javac Demo.java

生成文件如下:



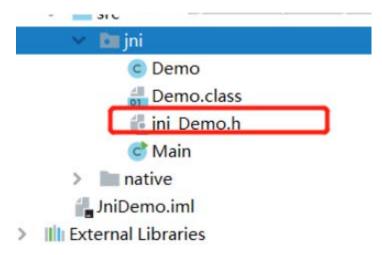
(3) 使用 javah 生成 h 文件:

需要注意的是:在 JDK10 之后,javah 命令已被移除,如果需要使用 javah 命令,可通过 javac -h 来实现。

这里命令如下:

```
javac -h ./ Demo.java
```

这是可以看到生成的 h 头文件如下:



h 文件中内容如下: 该文件自动生成,作文 C++文件和 Java 文件的关联。编译制作 dll 动态链接库需要用到,请勿修改。

```
2
    #include <jni.h>
 3
 4
 5
    #ifndef Included jni Demo
    #define _Included_jni_Demo
    #ifdef __cplusplus
    extern "C" {
    #endif
10
11
12
13
14
15
    JNIEXPORT void JNICALL Java_jni_Demo_sayHello
16
      (JNIEnv *, jobject, jint, jint);
17
18
   #ifdef __cplusplus
19
20
   #endif
21 #endif
```

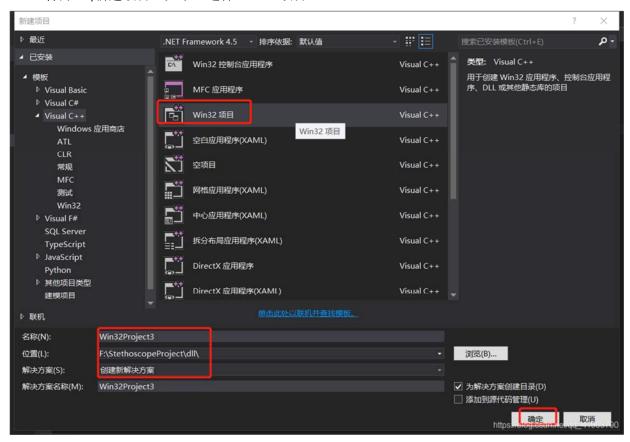
至此, Java 对应的接口方面内容完成。

2. 编译制作 dll 动态库

下面以 VS 来演示生成 dll 动态链接库文件。

2.1 创建 dll 工程

(1) 打开 VS,新建项目: 如下: 选择 Win32 项目:



(2) 选择 DLL 和空项目,如下:

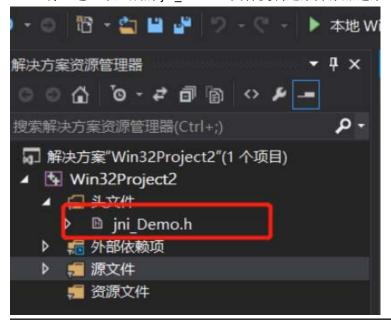
应用程序设置



新建完成。

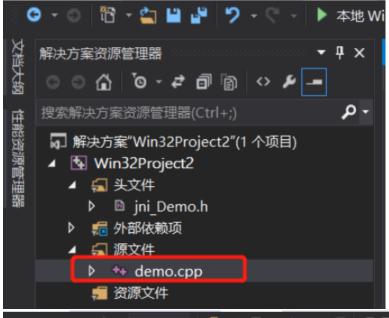
2.2 编写 C/C++源文件

(1) 将上述 1 中生成的 jni_Demo.h 文件复制过来并添加进来,如下:



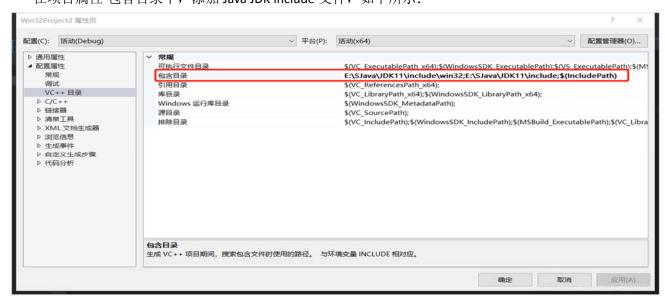
```
/indows 调试器 🔻 🔘 🔻 Debug 🔻 🎜 🚅 👺 🙉 🔚 🕰 🖺 🛂
jni Demo.h ⊅ ×
   (全局范围)
      1 /* DO NOT EDIT THIS FILE - it is machine generated */
          #include <jni.h>
         #ifndef _Included_jni_Demo
         #define _Included_jni_Demo
         #ifdef __cplusplus
          extern
                  #define cplusplus 199711L
          #endif
                      jni_Demo
          * Method:
                      sayHello
          * Signature: (II)V
          JNIEXPORT void JNICALL Java_jni_Demo_sayHello
          (JNIEnv *, jobject, jint, jint);
          #ifdef __cplusplus
          #endif
          #endif
```

(2) 编写接口实现 cpp 文件, 创建 demo.cpp 文件, 并编写代码实现:



(3) 配置包含目录

在项目属性-包含目录中,添加 Java JDK include 文件,如下所示:

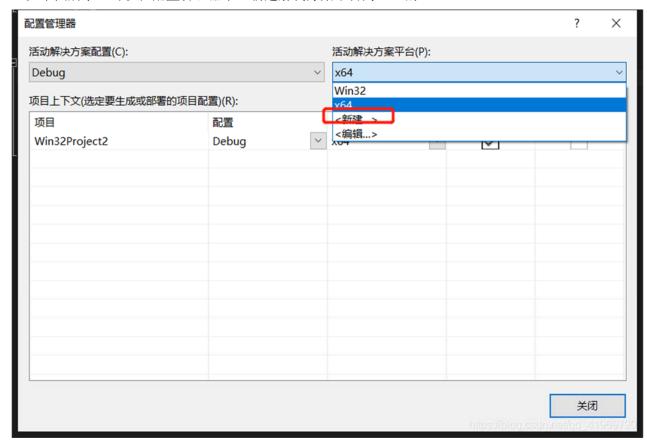


上面包含目录,需要修改为电脑上 JDK 安装路径中的 include 文件路径。

(4)(非必选)配置 dll 对应的系统版本

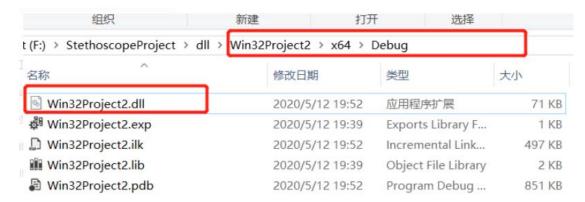
需要注意的是,生成的 dll 动态链接库,需要和系统位数对应, VS 默认是 32 位的,如果服务器是 64 位的,需要手动修改平台再生成。

如下图所示,可以在配置管理器中,新建解决方案平台为 X64 的:



2.3 生成 DLL 动态链接库文件

点击生成,即可在 Debug/Release 中生成 dll 文件。示例如下:



3. Java 程序调用

3.1 程序入口

(1) 新建程序入口类 Main 类,并在其中编写调用 C++代码的代码,程序示例如下:

```
public class Main {

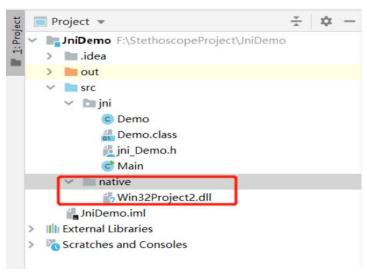
public static void main(String[] args){

System.load("F:\\StethoscopeProject\\JniDemo\\src\\native\\Win32Project2.dll");

Demo demo = new Demo();
demo.sayHello(2,3);
}
```

这里首先加载了 dll 动态链接库,后续通过调用 demo 对象的方法,即实现了 C++代码的调用。

(2) 将前述 2 生成的 dll 动态链接库文件放到程序的某路径下,并修改 load()中的文件路径为实际路径。如下:



```
© Demo.java × 💣 Main.java × 🛍 jni_Demo.h ×
      =/.../
       package jni;
       /** (一句话功能简述) <br> ...*/
       public class Main {
23
           public static void main(String[] args){
24
25
                System.load(filename: "F:\\StethoscopeProject\\JniDemo\\src\\mative\\Win32Project2.dl.")
26
27
                Demo demo = new Demo();
29
                demo.sayHello(x: 2, y: 3);
30
```

3.2 运行

启动程序,即可看到结果,这里输出为:

.encoding=UTF-8 -classpath F:\StethoscopeProject\JniDemo\out\production\JniDemo jni.Main x:2; y:3x+y= 5

Process finished with exit code 0

可以看到,程序加载了 C++的 dll 动态链接库,Java 应用可以正常调用 C++实现。