

博客

登录 | 注册

目录视图

摘要视图

RSS 订阅

huachao1001的专栏

每天学习一点点，每天进步一点点...

个人资料



huachao1001

关注

发私信

博客专家

访问：353521次

积分：3880

等级：BLOG > 5

排名：第8665名

原创：54篇

转载：1篇

译文：0篇

评论：375条

Android开发交流群



群名称：Android开发交流群

群号：340496711

博客专栏



AndroidStudio插件开发

文章：6篇

阅读：22796



Android OpenGL

文章：5篇

阅读：31946

Android学习

文章：8篇

异步赠书：10月Python畅销书升级

【线路图】人工智能到底学什么？！

程序员9月书讯

节后荐书：Python、PyQt5、Kotlin（评论送书）

原

IntelliJ IDEA平台下JNI编程（一）—HelloWorld篇

标签：JNI NDK IEDA Android Studio

2016-12-30 11:30

4767人阅读

评论(9)

微信关注CSDN 获得无限技术资源

快速回复

我要收藏

版权声明：本文为博主原创文章，未经博主允许不得转载。

目录(?)

[+]

转载请注明出处：【huachao1001的专栏：
<http://blog.csdn.net/huachao1001/article/details/53906237>】

JNI（Java Native Interface），出于学习JNI的目的，为了能够更方便快速地运行程序。本文的是在IDEA中进行，而不在AndroidStudio，这样能够对NDK的工作过程有个更深刻的认识，同时也能对JNI的原理有更深入的理解。虽然本文是HelloWorld篇，但是其中涉及到很多内容。博主将遇到的坑都记录下来了，希望能够帮到大家。这篇文章可能是2016年的最后一篇文章了，接下来JNI相关系列文章明年推出，欢迎大家关注。

1. 搭建GCC编译环境

既然使用的了JNI，那就不可避免地需要将C/C++文件编译成dll（windows）或so（Linux）文件。因为我是在Windows平台下开发，可以有如下选择：

1. 使用VC（或VS）编译成dll

2. 使用GCC编译成dll

因为开发Android应用肯定是需要编译成Linux平台的so文件，因此，为了后面开发Android程序的兼容，使用GCC编译器比较好。而Windows平台下的GCC又可以有如下选择：


1. 使用MinGW

2. 使用Cygwin

这里我选择了MinGW，不管选择哪个，只要能让本地有GCC编译环境即可。

http://blog.csdn.net/huachao1001/article/details/53906237

1/11



阅读：121481

文章分类

人工智能

(1)

Android

(44)

学习笔记

(9)

OpenGL

(5)

C语言

(1)

OpenCV

(1)

文章存档

2017年09月

(2)

2017年08月

(1)

2017年03月

(2)

2017年01月

(2)


2016年12月

(5)

展开

文章搜索

轻松一刻



点击即可启用 Adobe Flash Player

阅读排行

CoordinatorLayout的使用如...

(37235)

自定义View，有这一篇就够了

(31353)

酷炫的Activity切换动画，打...

(22730)

玩转AppBarLayout，更酷炫...

(20582)

从Android代码中来记忆23种...

(16823)

2017年秋季校招面经

(16239)

简单明了，彻底地理解Binder

(14326)

打造浪漫的Android表白程序

(12595)

打造属于你的LayoutManager

(11810)

Android OpenGL显示任意3...

(10215)

评论排行

2017年秋季校招面经

(53)

自定义View，有这一篇就够了

(35)

CoordinatorLayout的使用如...

(31)

简单明了，彻底地理解Binder

(28)

酷炫的Activity切换动画，打...

(23)

玩转AppBarLayout，更酷炫...

(23)

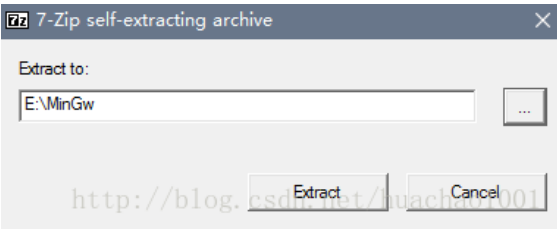
注意：搭建GCC编译环境时，一定要选择正确的GCC编译版本（32位和64位）。如果你本地安装的JDK是64位的，那么选择64位GCC，否则选择32位。这是为了使得编译后的库文件跟JVM的位一致，否则后面JVM无法调用dll（或so）。

1.1 安装MinGW

安装MinGW的方法很多，可以前往

<https://sourceforge.net/projects/mingw/files/MinGW/Base/gcc/> 中自己选择需
说，我也不是很清楚哪些包需要下载，没花时间研究，感兴趣的自己研究一些各个包
里有个安装教程<https://github.com/cpluspluscom/ChessPlusPlus/wiki/MinGW-Build-Tutorial>
按照这里的方法安装就可以。

另外，针对64位的，这里<https://nuwen.net/mingw.html>提供了完整的压缩包，直
<https://nuwen.net/files/mingw/mingw-14.1.exe>运行。如下：



其本质也是解压缩，把需要的MinGW库和程序解压到指定目录。如果懒的去看英文，我这里将我解
压后的重新压缩了下，大家去下载并且解压即可直接使用。

链接: <https://pan.baidu.com/s/1slpQrrJ>
密码: fykw

解压完成后，刚才指定的解压目录中的bin加入到path环境变量中。例如上面图中解压到
E:\MinGw，那么应当将E:\MinGw\bin加入到环境变量path中。注意，请确保bin目录确实在
E:\MinGw中，如果不在，可能在更深一层目录中，自行确定bin的目录。

做完后，打开控制台，输入：gcc -v，如下：

```
1 C:\Users\NetLab\Desktop>gcc -v
2 Using built-in specs.
3 COLLECT_GCC=gcc
4 COLLECT_LTO_WRAPPER=e:/mingw/bin/./libexec/gcc/x86_64-w64-mingw32/6.3.0/lto-wrapper.e
5 Target: x86_64-w64-mingw32
6 Configured with: ../src/configure --enable-languages=c,c++ --build=x86_64-w64-mingw32
7 Thread model: win32
8 gcc version 6.3.0 (GCC)
```

2. 开始编码

2.1 编写Java文件

新建一个 Java Project，创建包 com.huachao.java，如下：

- 打造属于你的LayoutManager (18)
- Android OpenGL 显示基本... (11)
- 打造浪漫的Android表白程序 (11)
- 在AndroidStudio中自定义Gr... (10)

最新评论

AndroidStudio插件开发（进阶篇之Acti...
qq_40339296 : 不错不错不错

自定义View，有这一篇就够了
alisa_fangtingting : 你好，请问第一个例子中android:layout_height="100dp"...
简单明了，彻底地理解Binder
qq_31013697 : @DGCP_PGM:照你这么...
说，为什么其他的进程通信机制需要两次，不也可以通过映射地址直接拿到数据，...

玩转AppBarLayout，更酷炫的顶部栏
alisa_fangtingting : 楼主，为什么我滑动的时候好卡，是怎么个情况

通过自定义Gradle插件修改编译后的class...
luyoulong123 : 正好研究一下

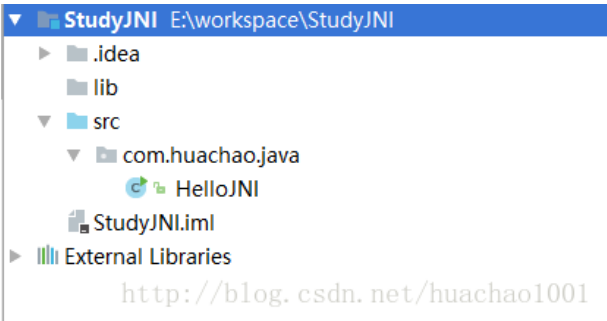
IntelliJ IDEA平台下JNI编程（一）—Hell...
qq_38545445 : 这个文章的确不错，查了这么多文章就这个感觉是好好写的。

简单明了，彻底地理解Binder
静默加载 : @DGCP_PGM:过了一年了，还有朋友回答太热心了

简单明了，彻底地理解Binder
MC_倾城 : @stven_king:关于Binder驱动可能楼主有些误解哦，Binder通过共享内存进行通信，只...

简单明了，彻底地理解Binder
MC_倾城 : 关于Binder驱动可能楼主有些误解哦，Binder通过共享内存进行通信，只有一次copy。Proc...

自定义View，有这一篇就够了
一小沫一 : @AA_chao:您好，请假下，您说的不应该是0，那为何还要说应该让currentHeight赋值为...



在包 `com.huachao.java` 下编写HelloJNI类：

```
1 package com.huachao.java;
2
3 /**
4  * Created by HuaChao on 2016/12/29.
5  */
6 public class HelloJNI {
7     static {
8         // hello.dll (Windows) or libhello.so (Unixes)
9         System.loadLibrary("hello");
10    }
11
12    private native void sayHello();
13
14    public static void main(String[] args) {
15
16        new HelloJNI().sayHello(); // invoke the native method
17    }
18
19 }
```

微信关注CSDN
获得无限技术资源

快速回复

我要收藏

函数System.loadLibrary()是加载dll（windows）或so（Linux）库，只需名称即可，无需加入文件名后缀（.dll或.so）。native关键字将函数sayHello()声明为本地函数，由C/C++实现。具体的实现就在hello.dll（Windows平台）或hello.so（Linux平台）中

2.2 生成JNI头文件

2.2.1 手动输入javah指令

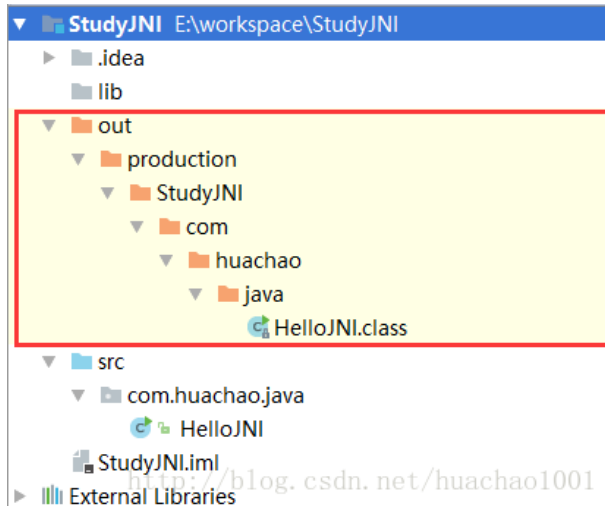
JNI生成头文件是通过JDK中提供的javah来完成，javah在 `{JDKHome}/bin` 目录中。用法如下：

```
1 javah -jni -classpath (搜寻类目录) -d (输出目录) (类名)
```

例如，将 `E:\Project\out\com\huachao\java` 目录中的 `HelloJNI.class` 生成头文件，并放入到 `E:\Project\jni` 中：

```
1 javah -jni -classpath E:\Project\out\com\huachao\java -d E:\Project\jni com.huachao.
```

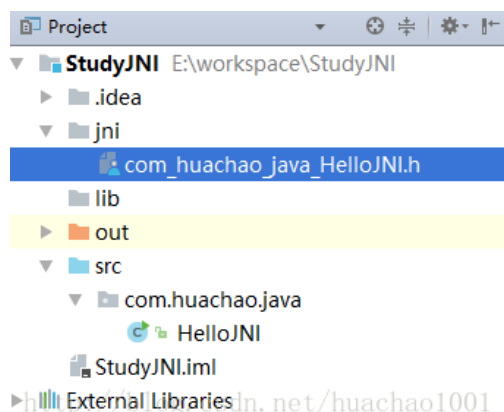
需要注意的是，使用javah来生成头文件（.h）时，-classpath指定的是编译后的java文件（.class）的目录，而不是源文件（.java）的目录，因此在使用javah指令之前，先build一下项目（或直接运行一下）。此时会生成out目录，所有编译后的文件都会存放在这个目录中。



接下来，直接在IDEA的Terminal窗口运行javah：



此时在jni目录中生成了头文件 com_huachao_java_HelloJNI.h



内容如下：

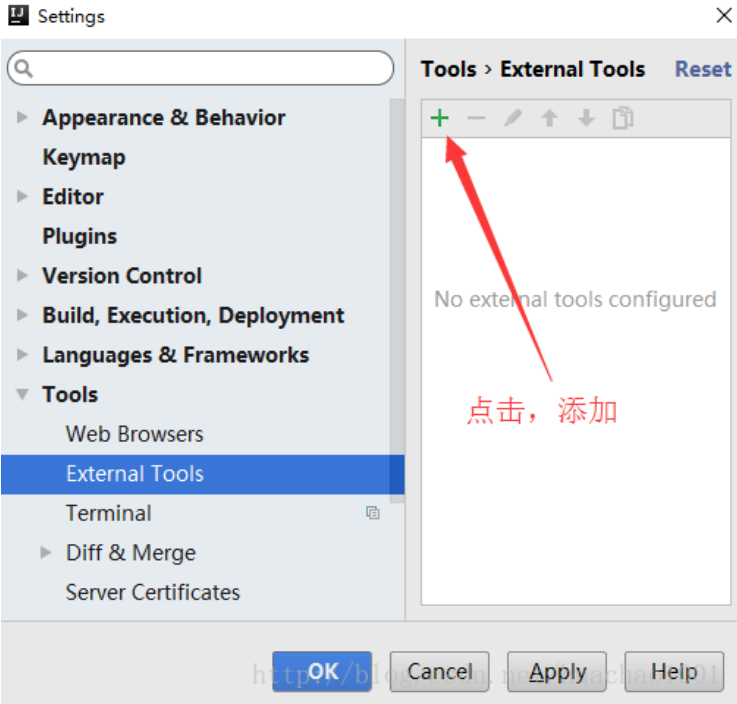
```
1  /* DO NOT EDIT THIS FILE - it is machine generated */
2  #include <jni.h>
3  /* Header for class com_huachao_java_HelloJNI */
4
5  #ifndef _Included_com_huachao_java_HelloJNI
6  #define _Included_com_huachao_java_HelloJNI
7  #ifdef __cplusplus
8  extern "C" {
9  #endif
10 /*
11  * Class:      com_huachao_java_HelloJNI
12  * Method:     sayHello
13  * Signature:  ()V
14  */
15 JNIEXPORT void JNICALL Java_com_huachao_java_HelloJNI_sayHello
16     (JNIEnv *, jobject);
17
18 #ifdef __cplusplus
19 }
```

```
20 #endif
21 #endif
```

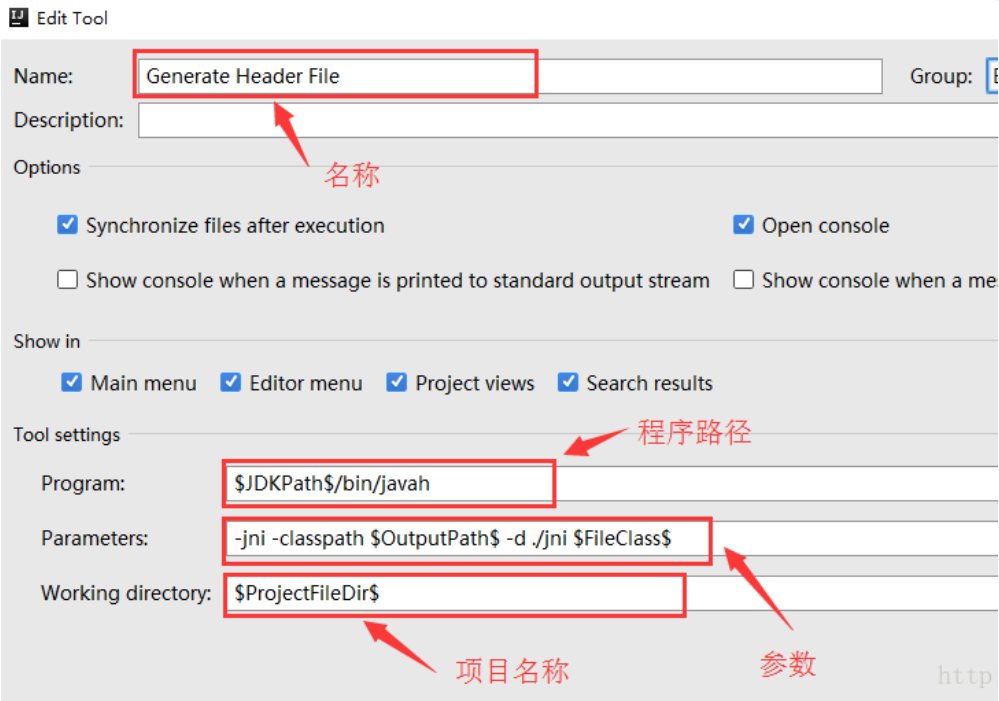
接下来我们只需实现 `Java_com_huachao_java_HelloJNI_sayHello(JNIEnv *, jobject)` 即可。仔细观察就会发现这个函数名称是有规律的，即 `Java_<包>_<类名>_<函数名>`，`JNIEXPORT`和`JNICALL`这两个宏定义暂时不用管。`JNIEnv` 和 `jobject`后面系列文章会详细介绍，这里暂时不理睬。

2.2.2 一键生成头文件

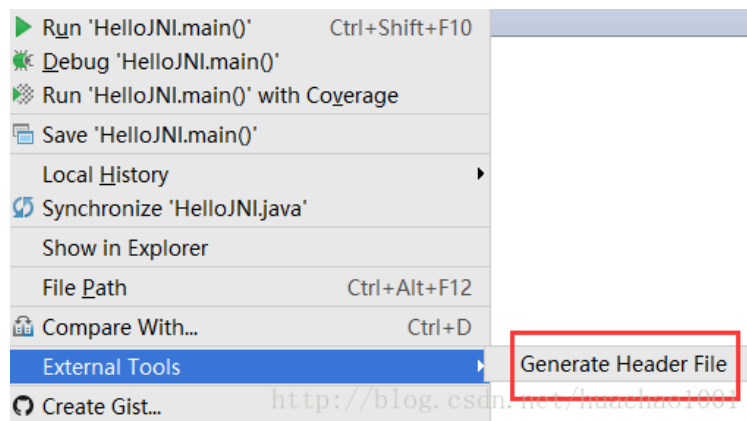
在2.2.1小节中，介绍了输入`javah`生成头文件方法。但是如果目录层次很深，或者是头文件的class文件，这工作量太大了，当然你可以通过写个小程序来实现。但是这里有一个更便捷的方法。点击 `File>Settings>Tools>External Tools`：



添加一个先的External Tools:



在HelloJNI.java文件中 点击右键>External Tools>Generate Header File ,



微信关注CSDN
获得无限技术资源

快速回复

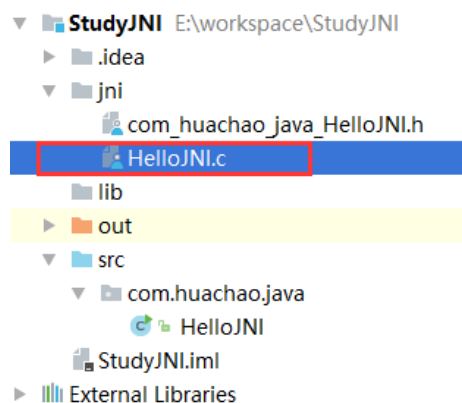
我要收藏

点击生成，可以看到Terminal窗口会自动运行指令。跟2.2.1小节的指令一模一样。

3. 编写C文件并编译成dll（或so）文件

3.1 手动输入命令生成

在jni目录中新建HelloJNI.c文件，如下：



编辑HelloJNI.c如下：

```
1 #include<jni.h>
2 #include <stdio.h>
3 #include "com_huachao_java_HelloJNI.h"
4
5 JNIEXPORT void JNICALL Java_com_huachao_java_HelloJNI_sayHello(JNIEnv *env, jobject th
6     printf("Hello World!\n");
7     return;
8 }
```

接下来就是使用GCC对HelloJNI.c编译，在Terminal窗口输入如下：

```
1 E:\workspace\StudyJNI>gcc -c jni/HelloJNI.c
2 jni/HelloJNI.c:1:17: fatal error: jni.h: No such file or directory
3 #include <jni.h>
```

发现报错，找不到jni.h头文件，将JDK目录中的include目录加入，即为：

```
1 E:\workspace\StudyJNI>gcc -c -I"E:\JDK\include" jni/HelloJNI.c
2 In file included from jni/HelloJNI.c:1:0:
3 E:\JDK\include\jni.h:45:20: fatal error: jni_md.h: No such file or directory
4 compilation terminated.
```

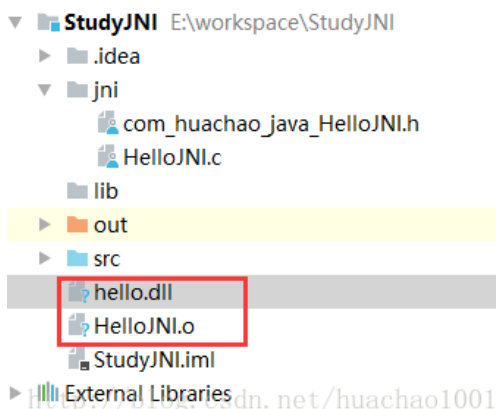
又报找不到jni_md.h错误，继续将JDK目录中的include/win32加入，即：

```
1 gcc -c -I"E:\JDK\include" -I"E:\JDK\include\win32" jni/HelloJNI.c
```

完成编译。此时在项目中会生成HelloJNI.o文件。接下来是将HelloJNI.o转为HelloJ windows平台下的动态链接库。在Terminal中输入如下：

```
1 gcc -Wl,--add-stdcall-alias -shared -o hello.dll HelloJNI.o
```

此时项目目录中生成了hello.dll文件：



3.2 一键生成dll

有了前面使用External Tools一键生成头文件的经验后，我们可以将编译成dll的过程命令也加入到External Tools中。前面将c文件编译链接成dll文件分了2个命令，这里我们直接通过一个命令来完成：

```
1 gcc -Wl,--add-stdcall-alias -I"E:\JDK\include" -I"E:\JDK\include\win32" -shared -o ./1
```

这样就将c文件编译成了dll，在这里把生成的dll文件加入到了lib目录中，而不是像之前那直接放到项目底下。因此在java.library.path应该指定目录为lib。

有了上面的命令后，可以很轻松的加入到External Tools中了。按照前面的方法，点

击 File>Settings>Tools>External Tools>+，输入内容如下：

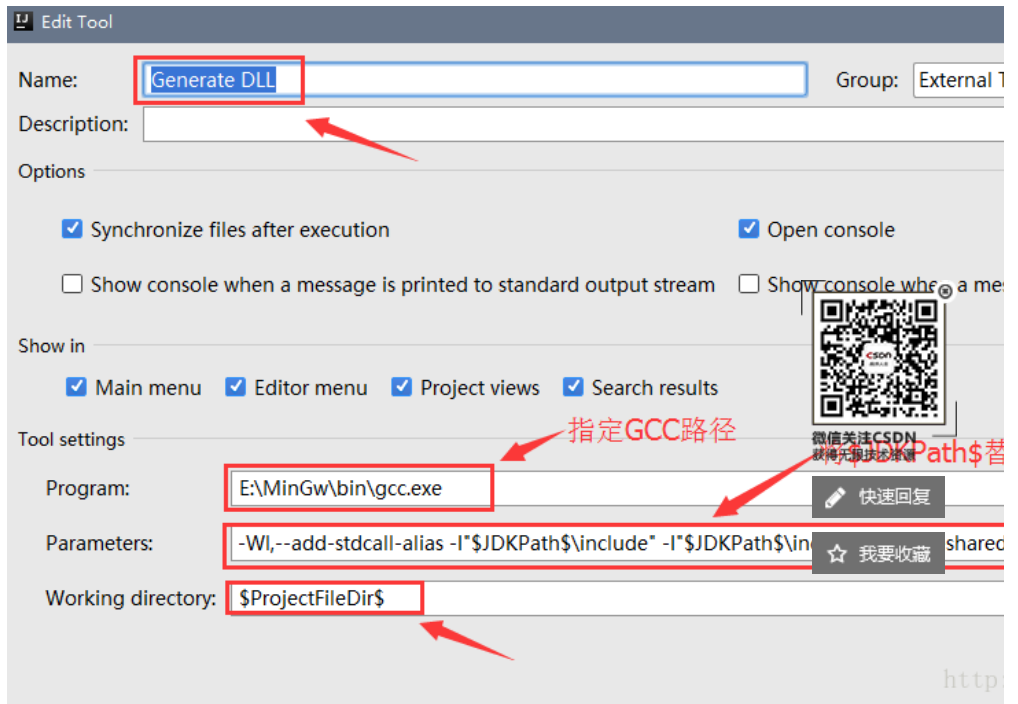
```
name : Generate DLL
Program : <GCC路径>
Parameters : -Wl,--add-stdcall-alias -I"%JDKPath%\include" -I"%JDKPath%\include\win32" -shared
Working Directory : %ProjectFileDir%
```



微信关注CSDN
获得无限技术资源

快速回复

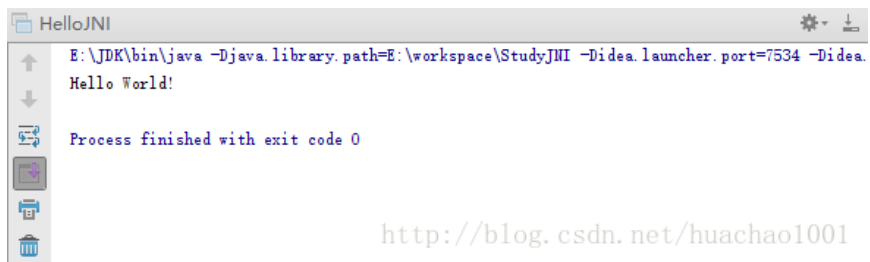
我要收藏



在HelloJNI.c中点击右键，选择External Tools>Generate DLL。此时，在lib目录中会得到dll文件。

4. 运行

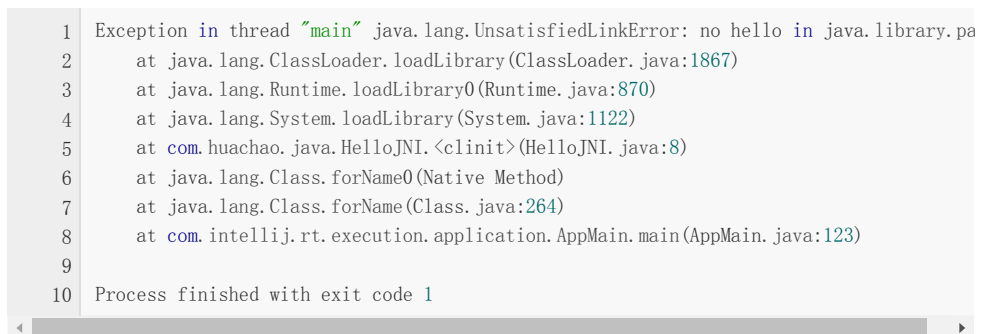
运行HelloJNI.java类后，如下：



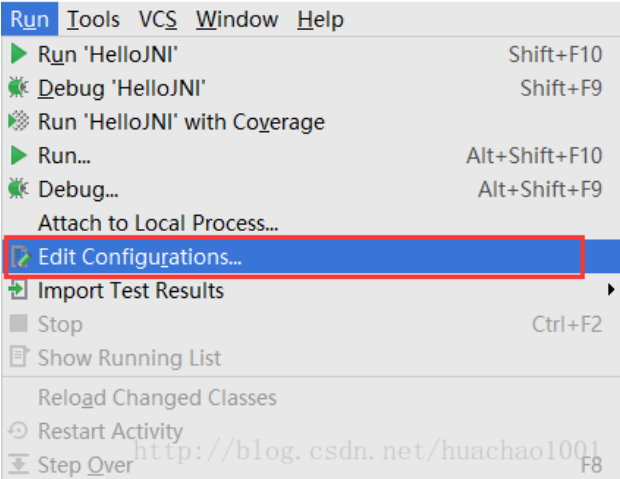
5 可能出现的错误

5.1 java.library.path找不到dll的错误

此时，点击直接运行HelloJNI.java类时，依然还会有错误：



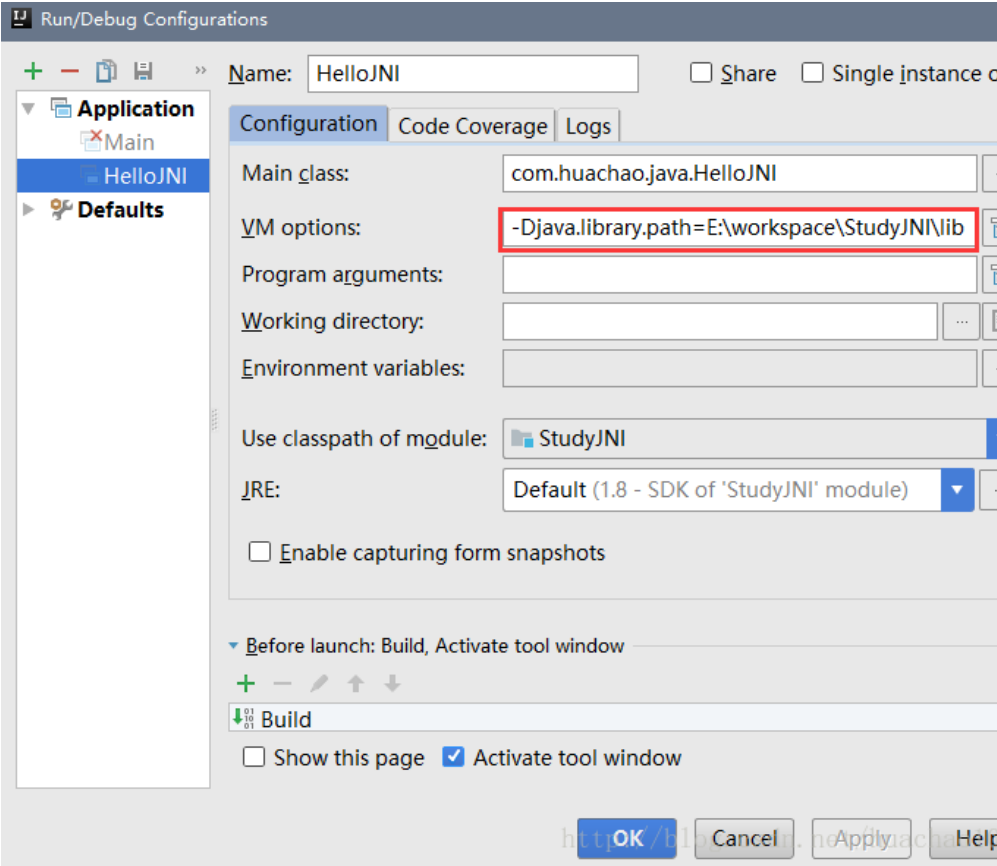
即找不到我们生成的dll文件。因为在Windows中JVM的 `java.library.path` 属性即为环境变量Path指定的目录，而我们生成的dll并未放入到Path指定的任何一个目录中，因此我们需要告诉JVM，dll文件在哪个目录中。点击 `Run > Edit Configurations...`，如下：



在VM options中加入java.library.path，指定dll（或so）文件所在的目录，比如本例为lib目录中的lib中，如下：

```
1 -Djava.library.path=E:\workspace\StudyJNI\lib
```

如下图所示：



5.2 无法识别__int64类型错误

错误如下：

```
1 error: unknown type name '__int64'
2 typedef __int64 jlong;
3 ^
```

出现这个错误的人一般是使用Cygwin GCC的人，这是因为Cygwin GCC不认识__int64类型，找到<JDK_HOME>/include/win32/jni_md.h，找到typedef__int64jlong;并修改为：

```
1 #ifdef __GNUC__
2     typedef long long jlong;
3 #else
4     typedef __int64 jlong;
5 #endif
```

或者是编译时将 __int64 加入，如下：

```
1 > gcc-3 -D __int64="long long" -mno-cygwin -Wl,--add-stdcall-alias
2 -I"<JAVA_HOME>\include" -I"<JAVA_HOME>\include\win32" -shared -o hel
```



微信关注CSDN
获得无限技术资源

快速回复

我要收藏

5.3 64-bit mode not compiled

错误如下：

```
1 HelloJNI.c:1:0: sorry, unimplemented: 64-bit mode not compiled in
2 #include <jni.h>
```

出现这个错误是因为，JDK版本是64位，而GCC编译器编译出的dll（或so）是32位，只需换个64位版本的GCC即可。

参考资料

- <https://www3.ntu.edu.sg/home/ehchua/programming/java/JavaNativeInterface.html#zz-3>
- <http://docs.oracle.com/javase/7/docs/technotes/guides/jni/spec/jniTOC.html>

顶

3

踩

0



- 上一篇 打造酷炫AndroidStudio插件
- 下一篇 IntelliJ IDEA平台下JNI编程（二）—类型映射

相关文章推荐

- JAVA IDE IntelliJ IDEA使用简介（一）—之界面...
- IntelliJ IDEA平台下JNI编程（三）—字符串、数组
- Presto服务治理与架构优化在京东的实践应用--王...
- 10小时深入掌握 Kubernetes
- IntelliJ IDEA使用笔记
- 打造浪漫的Android表白程序
- 【免费直播】Python最佳学习路线--韦玮
- JDK9新特性
- PAT Basic 1019. 数字黑洞 (20)
- IT求职经验总结——面试和准备策略
- JS-SDK开发与微信支付
- C++为什么不提倡使用scanf和printf函数
- C++ scanf()输入string类型变量
- C++最快的读取文件的方案(scanf,cin(及取消sync...
- Spring Cloud微服务真实场景实战解析
- C++ PAT - 1028. 人口普查(20)

查看评论



qq_38545445

8楼 2017-09-28 15:24发表

这个文章的确不错，查了这么多文章就这个感觉是好好写的。



kkopite

7楼 2017-08-27 22:32发表

一键生成dll的 参数一栏没有显示全呀大哥 ...
-Wl,--add-stdcall-alias -l"\$JDKPath\$include" -l"\$JDKPath\$include\$win32" -shared -o ./lib/\$FileNameWithoutExtension\$.dll ./jni/\$FileNameWithoutExtension\$.c



ll_gg_tt

6楼 2017-05-23

能把复杂的东西写这么清楚，很赞！！！（PS：我竟然看懂了）



StrayedKing

5楼 2017-04-03

好文要赞！

微信关注CSDN
获得无限技术资源

快速回复

我要收藏



NaiveCode

4楼 2017-03-10 16:22发表

问个问题：大型项目，.o文件怎么生成



jakck123

3楼 2017-01-06 09:52发表

你好，请问下将.o文件转成.so文件的命令行怎么写？



Android第捌卷

2楼 2017-01-03 21:38发表

知道了，谢谢你大哥。。。



Android第捌卷

1楼 2017-01-02 23:06发表

大哥你好，为什么 - 一键生成头文件 - 这个剪切图片，用的是android studio里面的啊。不是说 不再android studio环境下嘛？



huachao1001

Re: 2017-01-03 08:21发表

回复Android第捌卷：你好，那个不是AndroidStudio里，而是IntelliJ IDEA，AndroidStudio 是基于IntelliJ IDEA的

您还没有登录,请[登录](#)或[注册](#)

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

[公司简介](#) | [招贤纳士](#) | [广告服务](#) | [联系方式](#) | [版权声明](#) | [法律顾问](#) | [问题报告](#) | [合作伙伴](#) | [论坛反馈](#)

网站客服 杂志客服 微博客服 webmaster@csdn.net 400-660-0108 | 北京创新乐知信息技术有限公司 版权所有 | 江苏知之为计算机有限公司 |

江苏乐知网络技术有限公司

京 ICP 证 09002463 号 | Copyright © 1999-2017, CSDN.NET, All Rights Reserved