[C]C 语言用 JVM 调起 Java 方法

作者：lin772662623
原文链接：https://ld246.com/article/1530184740674
来源网站：链滴

<img src="https://ld246.com/images/img-loading.svg" alt="" data-src="https://b3logfile
com/bing/20180623.jpg?imageView2/1/w/960/h/520/interlace/1/q/100">
一边重温写过的文章，一边整理到这边来。
<hr>
有个纯 C 项目，需要调用第三方的库，但是第三方目前只有 Java 的库，所以我只能在全 C 项目
想办法调用这个 JAVA 库， 那么这个时候就考虑用到 libjvm.so 来创建 JVM 来运行类。
<hr>
<h3 id="1-在linux环境下安装jdk-在jdk中-包含以下文件--">1.在 linux 环境下安装 jdk，在 jdk 中
包含以下文件： </h3>
/usr/local/jdk1.8.0_131/include/jnk.h
/usr/local/jdk1.8.0_131/include/linux/jni_md.h
/usr/local/jdk1.8.0_131/jre/lib/i386/server/libjvm.so
<hr>
<h3 id="2-编写需要被C调用的java类">2.编写需要被 C 调用的 java 类</h3>
<pre><code class="language-go highlight-chroma"><span cla
s="highlight-cl">package t
st;
<span class="high
ight-nx">public class <span class="highlight-n
">HelloWorld {
    <span class=
highlight-nx">public static <span class="highli
ht-kt">int intMethod<span class="highlight-p"
(int n<spa
 class="highlight-p">){
        <span c
ass="highlight-k">return n<span class="highli
ht-o">*n;
    <span class=
highlight-p">}

    <span class=
highlight-nx">public static <span class="highli
ht-nx">boolean booleanMethod<span class="h
ghlight-p">(boolean <span class="highlight-kt
>bool) {
        <span c
ass="highlight-k">return !<span class="highligh
-kt">bool;
    <span class=
highlight-p">}
    <span class=
highlight-nx">public static <span class="highli
ht-nx">void sayHello<span class="highlight-p"
()
    <span class=
highlight-p">{
        <span c
ass="highlight-nx">System.<span class="highli
ht-nx">out.println
/span>("say Hello from C."</
pan>);
    <span class=
highlight-p">}

<span class="high
ight-p">}
</code></pre>
里面有三个静态函数，分别为不同返回值，不同参数，
<hr>
<h3 id="3-开始编写创建调用JVM的C代码了-">3.开始编写创建调用 JVM 的 C 代码了。</h3>
<pre><code class="highlight-chroma"><span class="highlight
cl">#include <jni.h>
//jni.h文件包含在 C
代码中所需要的 JNI 的所有类型和函数定义
#ifdef _WIN32
#define PATH_SE
ARATOR ';'
#else
#define PATH_SE
ARATOR ':'
#endif
//1.包括准备本机
用程序以处理 Java 代码
//2.将 JVM 嵌入
机应用程序
//3.然后从该应用
序内找到并调用 Java 方法。
int main()
{
/*
接下来，声明所有
望在程序中使用的变量。
JavaVMOption opt
ons[] 具有用于 JVM 的各种选项设置。
当声明变量时，确
所声明的JavaVMOption options[] 数组足够大，以便能容纳您希望使用的所有选项。
在本例中，我们使
的唯一选项就是类路径选项。
因为在本示例中，
们所有的文件都在同一目录中，所以将类路径设置成当前目录。
可以设置类路径，
它指向任何您希望使用的目录结构。*/
    JavaVMOption
ptions[1];
    JNIEnv *env;
    JavaVM *jvm;
    JavaVMInitArgs
vm_args;
/*JNIEnv *env
 表示 JNI 执行环境。
JavaVM jvm
 是指向 JVM 的指针,我们主要使用这个指针来创建、初始化和销毁 JVM。
JavaVMInitArgs v
_args 表示可以用来初始化 JVM 的各种 JVM 参数。*/

    long status;
    jclass cls;
    jmethodID mid;

```
</span></span><span class="highlight-line"><span class="highlight-cl">    jint square;
</span></span><span class="highlight-line"><span class="highlight-cl">    jboolean not;
</span></span><span class="highlight-line"><span class="highlight-cl">/*avaVMInitArgs
构表示用于 JVM 的初始化参数。
</span></span><span class="highlight-line"><span class="highlight-cl">在执行 Java 代码
前，可以使用这些参数来定制运行时环境。
</span></span><span class="highlight-line"><span class="highlight-cl">正如您所见，这些
项是一个参数,而 Java 版本是另一个参数。
</span></span><span class="highlight-line"><span class="highlight-cl">按如下所示设置了
些参数：*/
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">/*为 JVM 设置类
径，以使它能找到所需要的 Java 类。
</span></span><span class="highlight-line"><span class="highlight-cl">在这个特定示例中
因为 Sample2.class 和Sample2.exe 都位于同一目录中，所以将类路径设置成当前目录。
</span></span><span class="highlight-line"><span class="highlight-cl">我们用来为 Sampl
2.c 设置类路径的代码如下所示：*/
</span></span><span class="highlight-line"><span class="highlight-cl">    options[0].opti
nString = "-Djava.class.path=.";
</span></span><span class="highlight-line"><span class="highlight-cl">    memset(&amp;
m_args, 0, sizeof(vm_args));
</span></span><span class="highlight-line"><span class="highlight-cl">    vm_args.version
= JNI_VERSION_1_2;
</span></span><span class="highlight-line"><span class="highlight-cl">    vm_args.nOpti
ns = 1;
</span></span><span class="highlight-line"><span class="highlight-cl">    vm_args.option
 = options;
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">/*创建 JVM
</span></span><span class="highlight-line"><span class="highlight-cl">处理完所有设置之
，现在就准备创建 JVM 了。先从调用方法开始
</span></span><span class="highlight-line"><span class="highlight-cl">如果成功，则这个
法返回零，否则，如果无法创建 JVM，则返回JNI_ERR。*/
</span></span><span class="highlight-line"><span class="highlight-cl">    status = JNI_Cr
ateJavaVM(&amp;jvm, (void**)&amp;env, &amp;vm_args);
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    if (status != JNI
ERR)
</span></span><span class="highlight-line"><span class="highlight-cl">    {
</span></span><span class="highlight-line"><span class="highlight-cl">/*
</span></span><span class="highlight-line"><span class="highlight-cl">查找并装入 Java 类
</span></span><span class="highlight-line"><span class="highlight-cl">一旦创建了 JVM
后，就可以准备开始在本机应用程序中运行 Java 代码。
</span></span><span class="highlight-line"><span class="highlight-cl">首先，需要使用Fin
Class() 函数查找并装入 Java 类，如下所示：
</span></span><span class="highlight-line"><span class="highlight-cl">cls 变量存储执行Fi
dClass() 函数后的结果,如果找到该类，则 cls 变量表示该Java 类的句柄,
</span></span><span class="highlight-line"><span class="highlight-cl">如果不能找到该类
则 cls 将为零。
</span></span><span class="highlight-line"><span class="highlight-cl">*/
</span></span><span class="highlight-line"><span class="highlight-cl">        cls = (*env)-
gt;FindClass(env, "test/HelloWorld");
</span></span><span class="highlight-line"><span class="highlight-cl">        printf("test1,c
s=%d...\n",cls);
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">        if(cls !=0)
</span></span><span class="highlight-line"><span class="highlight-cl">        {
</span></span><span class="highlight-line"><span class="highlight-cl">/*
</span></span><span class="highlight-line"><span class="highlight-cl">查找 Java 方法
</span></span><span class="highlight-line"><span class="highlight-cl">接下来，我们希望 GetStaticMethodID() 函数在该类中查找某个方法。
</span></span><span class="highlight-line"><span class="highlight-cl">我们希望查找方法 itMethod，它接收一个 int 参数并返回一个 int。
</span></span><span class="highlight-line"><span class="highlight-cl">以下是查找 intMethod 的代码：
</span></span><span class="highlight-line"><span class="highlight-cl">*/
</span></span><span class="highlight-line"><span class="highlight-cl">            mid = (*env)-&gt;GetStaticMethodID(env, cls, "intMethod", "(I)I");
</span></span><span class="highlight-line"><span class="highlight-cl">/*
</span></span><span class="highlight-line"><span class="highlight-cl">mid 变量存储执行 GetStaticMethodID() 函数后的结果。
</span></span><span class="highlight-line"><span class="highlight-cl">如果找到了该方法则 mid 变量表示该方法的句柄。
</span></span><span class="highlight-line"><span class="highlight-cl">如果不能找到该方法，则mid 将为零。
</span></span><span class="highlight-line"><span class="highlight-cl">*/
</span></span><span class="highlight-line"><span class="highlight-cl">            if(mid !=0)
</span></span><span class="highlight-line"><span class="highlight-cl">            {
</span></span><span class="highlight-line"><span class="highlight-cl">/*CallStaticIntMethod() 方法接受 cls（表示类）、mid（表示方法）以及用于该方法一个或多个参数。
</span></span><span class="highlight-line"><span class="highlight-cl">在本例中参数是 int 5。*/
</span></span><span class="highlight-line"><span class="highlight-cl">                square = (*env)-&gt;CallStaticIntMethod(env, cls, mid, 5);
</span></span><span class="highlight-line"><span class="highlight-cl">                printf("Result of intMethod: %d\n", square);
</span></span><span class="highlight-line"><span class="highlight-cl">            }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">            mid = (*env)-&gt;GetStaticMethodID(env, cls, "booleanMethod", "(Z)Z");
</span></span><span class="highlight-line"><span class="highlight-cl">            if(mid !=0)
</span></span><span class="highlight-line"><span class="highlight-cl">            {
</span></span><span class="highlight-line"><span class="highlight-cl">                not = (*env)-&gt;CallStaticBooleanMethod(env, cls, mid, 1);
</span></span><span class="highlight-line"><span class="highlight-cl">                printf("Result of booleanMethod: %d\n", not);
</span></span><span class="highlight-line"><span class="highlight-cl">            }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">            mid = (*env)-&gt;GetStaticMethodID(env, cls, "sayHello", "()V");
</span></span><span class="highlight-line"><span class="highlight-cl">            if(mid !=0)
</span></span><span class="highlight-line"><span class="highlight-cl">            {
</span></span><span class="highlight-line"><span class="highlight-cl">                (*env)-&gt;CallStaticVoidMethod(env, cls, mid);
</span></span><span class="highlight-line"><span class="highlight-cl">                printf("Result of voidMethod");
</span></span><span class="highlight-line"><span class="highlight-cl">            }
</span></span><span class="highlight-line"><span class="highlight-cl">        }
```

原文链接：[C]C 语言用 JVM 调起 Java 方法

        (*jvm)->D
stroyJavaVM(jvm);
        return 0;
    }
    else
    return -1;
}
</code></pre>
对于 GetStaticMethodID 最后一个参数：
signature，用字符串是描述了函数的参数和返回值。
"()V"
"(II)V"
"(Ljava/lang/String;Ljava/lang/String;)V"
实际上这些字符是与函数的参数类型一一对应的。
"()" 中的字符表示参数，后面的则代表返回值。例如"()V" 就表示 void Func();
"(II)V" 表示 void Func(int, int);
那其他情况呢？请查看下表：
<table>
<thead>
<tr>
<th>类型</th>
<th>符号</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>Z</td>
</tr>
<tr>
<td>byte</td>
<td>B</td>
</tr>
<tr>
<td>char</td>
<td>C</td>
</tr>
<tr>
<td>short</td>
<td>S</td>
</tr>
<tr>
<td>int</td>
<td>I</td>
</tr>
<tr>
<td>long</td>
<td>L</td>
</tr>
<tr>
<td>float</td>
<td>F</td>
</tr>

<tr>
<td>double</td>
<td>D</td>
</tr>
<tr>
<td>void</td>
<td>V</td>
</tr>
<tr>
<td>object 对象</td>
<td>LClassName; L 类名;</td>
</tr>
<tr>
<td>Arrays</td>
<td>[array-type [数组类型</td>
</tr>
<tr>
<td>methods 方法</td>
<td>(argument-types)return-type (参数类型)返回类型</td>
</tr>
<tr>
<td><a href="https://ld246.com/forward?goto=http%3A%2F%2Fzhiweiofli.iteye.com%2Fblo%2F1830321" target="_blank" rel="nofollow ugc">具体可以看这里，引用自 zhiweiofli 的博客</></td>
<td></td>
</tr>
</tbody>
</table>
<code>话说是solo自带的markdown编辑器，和csdn语法还不一样呢。。 是不是要考虑换个Editor</code>
<hr>
<h3 id="4-编译">4.编译</h3>
写好了 java 和 C 的代码之后，就开始编译了。
<pre><code class="highlight-chroma">javac HelloWorld.java
</code></pre>
将编译好的 HelloWorld.class 放到当前目录的 test/HelloWorld.class 下，接下来编译 C，
<pre><code class="highlight-chroma">gcc -o Hello Hello.c -D_JNI_IMPLEMENTATION_ -I/usr/local/jdk1.8.0_131/include/ -I/usr/local/jdk1.8.0_131/include/linux/ -L/usr/local/jdk1.8.0_131/jre/lib/i386/server/ -ljvm
</code></pre>
<h3 id="5-运行">5，运行</h3>
<pre><code class="highlight-chroma">./Hello
</code></pre>
运行结果：
test1,cls=142029704...
Result of intMethod: 25
Result of booleanMethod: 0
say Hello from C.