Android 框架基础 -JNI

作者：CM
原文链接：https://ld246.com/article/1486375868036
来源网站：链滴

1.NDK 与 JNI 的区别：
NDK 是为了方便开发基于 JNI 的应用而提供的一套开发和编译工具集；
JNI 是一套编程接口，可以运用在应用层，应用框架层，用以实现 Java 代码与本地代码的交互。
2.编程的模型结构：
第一步：Java 层声明 Native 方法。
第二步：JNI 层实现 Java 层声明的 Native 方法，在 JNI 层可以调用底层库或者毁掉 Java 层方。
第三步：加载 JNI 层代码编译后生成的共享库。
JNI 中的 JNIEnv 是什么：
在 C++ 中：JNIEnv 就是 struct_JNIEnv。JNIEnv* env 等价于 struct _JNIEvn* env,在调用 JNI 函数的时候，只需要 env->FindClass(JNIEnv*,const char*)，就会间接调用 JNINativeInterface 构体里定义的函数指针，而无需先对 env 解引用。
在 C 中：JNIEnv 就是 const struct JNINativeInterface*.JNIEnv* env 实际等价于 const struct NINativeInterface** env，因此要先解引用得到 env，才能调用到 JNINativeInterface 结构里面定义的函数指针。解引用调用方法即(env)->FindClass(JNIEnv,const char</e>)。
注意：JNIEnv 只在当前现成合纵有笑。本地方法不能将 JNIEnv 从一个现成传递到另一个现成。相同的 java 现成中对本地方法多次调用的时候，传递给该本地方法的 JNIEnv 是相同的。但是一个地方法可以被不同的 Java 现成所调用，因此可以接受不同的 JNIEnv。
第一节 在 Java 中调用 JNI 实现方法
Java 数据与 JNI 数据类型转换。
（Java 层调用 JNI 方法传递参数是 Java 类型需要通过 dalvik 虚拟机转换后才能被 JNI 识别。）
Java-JNI 基本数据类型转换关系表
| Java 类型 | JNI 类型 | 字 长 |
| boolean | jboolean | 8 位 |
| byte | jbyte | 8 位 |
| char | jchar | 16 位 |
| short | jshort | 16 位 |
| int | jint | 32 位 |
| long | jlong | 64 位 |
| float | jfloat | 32 位 |
| double | jdouble | 64 位 |
| void | void | |
| | | |
JNI 方法命名规则：
Java_前缀
全限定的类名
下划线（_）分隔符
增加第一参数 JNIEnv* env
增加第二个参数 jobject
其他参数按类型映射
返回值按类型映射
例如：
public static native boolean isLoggable(String tag, int level);
对应的 JNI 方法实现：
static jboolean android_util_Log_isLoggable(JNIEnv* env,jobject clazz,jstring tag,jint level)
//...实现代码};
在框架层中可以采用函数注册的方式建立 Java 层生命方法与 JNI 层实现方法之间的对应关系，以不遵守这些规则。
JNI 方法签名规则：（用一个字符来识别重载方法）
<hr>

JNI 类型签名规则
| Java 类型 | 类型签名 | Java 类型 | 类型签名 |
| boolean | Z | long | J |
| byte | B | float | F |
| char | C | double | D |
| short | S | 类 | L 全限定类名; |
| int | I | 数组 | [元素类型签名 |
例如：
long fun(int n , String str,int[] arr);
对应的方法签名：
(ILjava/lang/String;[I)J
JNI 操作 Java 对象（通过操作 jobject 带来操作 Java 对象提供的变量和方法）：
FindClass 和 GetObjectClass
在 C++ 中：
jclass FindClass(const char* name);//查找类信息
jclass GeObjectClass(jobject obj);//返回对象的类
在 C 中：
jclass （FindClass)(JNIEnv ,const char*);
jclass (GetObjectClass)(JNIEnv , jobject);
注意：通过 FindClass 传入要查找的类的全限定名（以“/”分隔路径）即可，之后方法返回一个 class 对象，这样就可以操作这个类的方法和变量了。
操作成员变量（域）和方法：
Java 中习惯将变量成为成员变量，而不是域。为了兼容 JNI 命名规则和 Java 习惯，将域和变量价。
操作变量=》以 Log 系统为例：
第一步：通过 FindClass 方法找到类的信息 clazz；
第二步：以 clazz 为参数调用 GetStaticFieldID(clazz,"DEBUG","I");其中 DEBUG 是要访问的 Java 域的名字，I 是该域的类型签名，即整型。
第三步：GetStaticFieldID 函数返回一个 jfieldID，代表 Java 成员变量。最后将该 ID 传给 GetStaticField 方法，得到 Java 层的成员变量 DEBUG 的值。
操作方法：
与操作成员变量类似，流程如下：
FindClass->GetMethodID(返回 jmethodID)->CallMethod
其中 Type 为方法的修饰符，如 public ,private 等等