JNI数据结构之JNINativeMethod

1、JNINativeMethod 结构体的官方定义

typedef struct {
const char* name;
const char* signature;
void* fnPtr;
} JNINativeMethod;
第一个变量name是Java中函数的名字。
第二个变量signature，用字符串是描述了Java中函数的参数和返回值
第三个变量fnPtr是函数指针，指向native函数。前面都要接 (void *)
第一个变量与第三个变量是对应的，一个是java层方法名，对应着第三个参数的native方法名字

```
/*
 * 由于gMethods[]是一个<名称，函数指针>对照表，在程序执行时，
 * 可多次调用registerNativeMethods()函数来更换本地函数的指针，
 * 从而达到弹性调用本地函数的目的。
 *具体可以参看http://blog.csdn.net/conowen/article/details/7521340
 */
static JNINativeMethod gMethods[] = {
    {"setDataSource",       "(Ljava/lang/String;)V",          (void
*)com_media_ffmpeg_FFMpegPlayer_setDataSource},
    {"_setVideoSurface",    "(Landroid/view/Surface;)V",      (void
*)com_media_ffmpeg_FFMpegPlayer_setVideoSurface},
    {"prepare",             "()V",                            (void
*)com_media_ffmpeg_FFMpegPlayer_prepare},
    {"_start",              "()V",                           (void
*)com_media_ffmpeg_FFMpegPlayer_start},
    {"_stop",               "()V",                            (void
*)com_media_ffmpeg_FFMpegPlayer_stop},
    {"getVideoWidth",       "()I",                            (void
*)com_media_ffmpeg_FFMpegPlayer_getVideoWidth},
    {"getVideoHeight",      "()I",                            (void
*)com_media_ffmpeg_FFMpegPlayer_getVideoHeight},
    {"seekTo",              "(I)V",                           (void
*)com_media_ffmpeg_FFMpegPlayer_seekTo},
    {"_pause",              "()V",                            (void
*)com_media_ffmpeg_FFMpegPlayer_pause},
    {"isPlaying",           "()Z",                            (void
*)com_media_ffmpeg_FFMpegPlayer_isPlaying},
    {"getCurrentPosition",  "()I",                            (void
```

```
*)com_media_ffmpeg_FFMpegPlayer_getCurrentPosition},
    {"getDuration",         "()I",                          (void
*)com_media_ffmpeg_FFMpegPlayer_getDuration},
    {"_release",            "()V",                          (void
*)com_media_ffmpeg_FFMpegPlayer_release},
    {"_reset",              "()V",                          (void
*)com_media_ffmpeg_FFMpegPlayer_reset},
    {"setAudioStreamType",  "(I)V",                         (void
*)com_media_ffmpeg_FFMpegPlayer_setAudioStreamType},
    {"native_init",         "()V",                          (void
*)com_media_ffmpeg_FFMpegPlayer_native_init},
    {"native_setup",        "(Ljava/lang/Object;)V",        (void
*)com_media_ffmpeg_FFMpegPlayer_native_setup},
    {"native_finalize",     "()V",                          (void
*)com_media_ffmpeg_FFMpegPlayer_native_finalize},
    {"native_suspend_resume", "(Z)I",                       (void
*)com_media_ffmpeg_FFMpegPlayer_native_suspend_resume},
};
```

主要是第二个参数比较复杂：

括号里面表示参数的类型，括号后面表示返回值。

"()" 中的字符表示参数，后面的则代表返回值。例如"()V" 就表示void Fun();
"(II)V" 表示 void Fun(int a, int b);
这些字符与函数的参数类型的映射表如下：

2、第二个参数之基本数据类型

Field Descriptor	Java Language Type
Z	boolean
B	byte
C	char
S	short
I	int
J	long
F	float
D	double

3、第二个参数之对象类型与数组类型

Field Descriptor	Java Language Type
"Ljava/lang/String;"	String
"[I"	int[]
"[Ljava/lang/Object;"	Object[]

对象类型：以"L"开头，以";"结尾，中间是用"/" 隔开。如上表第1个
数组类型：以"["开始。如上表第2个（n维数组的话，则是前面多少个"["而已，如"[[[D"表示“double[][][]”）
对象数组类型：上述两者结合，如上表第3个

3.1、对象类型与数组类型的举例：

Method Descriptor	Java Language Type
"()Ljava/lang/String;"	String f();
"(ILjava/lang/Class;)J"	long f(int i, Class c);
"([B)V"	String(byte[] bytes);