

## 管满满

热爱生活，热爱编程，分享源于学习。

博客园 首页 新随笔 联系 订阅 管理

# Android NDK开发之C调用Java及原生代码断点调试（二）

上一篇中，我们主要学习了Java调用本地方法，并列举了两大特殊实例来例证我们的论据，还没学习的伙伴必须先去阅读下，本次的学习是直接在上一篇的基础上进行了。点击：[Android NDK开发之从Java与C互调中详解JNI使用（一）](#)

本篇我们主要学习如何从C源码中调用Java代码，以及使用gradle-experimental来调试原生代码。

## C 调用 Java 成员变量

1. 首先我们现在Java2CJNI类中定义几个成员变量，如下：

### 公告

昵称：管满满  
园龄：1年6个月  
粉丝：13  
关注：4  
[+加关注](#)

|              |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|
| < 2017年12月 > |    |    |    |    |    |    |
| 日            | 一  | 二  | 三  | 四  | 五  | 六  |
| 26           | 27 | 28 | 29 | 30 | 1  | 2  |
| 3            | 4  | 5  | 6  | 7  | 8  | 9  |
| 10           | 11 | 12 | 13 | 14 | 15 | 16 |
| 17           | 18 | 19 | 20 | 21 | 22 | 23 |
| 24           | 25 | 26 | 27 | 28 | 29 | 30 |
| 31           | 1  | 2  | 3  | 4  | 5  | 6  |



这里定义了两个普通成员变量和一个静态成员变量。

就像C不能直接使用Java的引用类型一样，C也不能直接的访问Java成员变量，而是通过JNI所封装的API来调用Java成员。通常会有如下的步骤：

- 1：获取java实例对象的引用
- 2：通过实例对象获取java成员变量ID
- 3：通过变量ID获取java成员变量

那么我们现在分步的讲下学习以上的步骤。

## 获取java实例对象的引用

获取实例对象的引用JNI已为我们封装好了方法，我们可以使用GetObjectClass函数来获取class对象：

```
jclass (GetObjectClass)(JNIEnv, jobject);
```

例如：

```
jclass class = (*env)->GetObjectClass(env, jobj);
```

jobj对象我们在上一节也讲过，这个是Java调用本地方法时，JNI会封装调用类的一个实例，在这里就是Java2CJNI类的引用。

搜索

找找看

谷歌搜索

我的标签

Android(11)

5.0新特性(3)

RecyclerView(3)

源码分析(3)

自定义View(3)

view(2)

JNI(2)

NDK(2)

React Native(1)

6.0新特性(1)

另外一种方法也是可以获取到class对象，就是通过反射机制来获取对象：

jclass (FindClass)(JNIEnv, const char\*);

例如：

```
jclass class = (*env)->FindClass(env, "com/sanhui/ndkdemo/Java2CJNI");
```

同上面的方法一样，都可以获取Java对象引用。

## 通过实例对象获取java成员变量ID

由第一步我们获取到了实例对象的引用，那么我们可以通过JNI封装的方法来获取实例内的变量ID：

①：获取普通成员变量ID

通过jfieldID (GetFieldID)(JNIEnv, jclass, const char, const char);可以获取到一个jfieldID 类型的ID。

GetFieldID函数中第三个参数是Java类中的成员变量的名称，如果在Java2CJNI类中定义的成员private String codeError = "验证码错误!"中codeError。第四个参数是变量签名，说白了就是Java类中成员变量的返回类型，如变量codeError 的返回类型是String ，但是String在原生代码中属于引用类型，不能直接识别，所以在JNI中有相应的签名映射，如下表：

| Java 类型               | JNI 签名映射                 |
|-----------------------|--------------------------|
| boolean               | Z                        |
| byte                  | B                        |
| char                  | C                        |
| short                 | S                        |
| int                   | I                        |
| long                  | J                        |
| float                 | F                        |
| double                | D                        |
| fully-qualified-class | Lfully-qualified-class ; |
| type[]                | [type                    |
| method type           | ( arg-types ) ret-type   |

更多

随笔档案

2017年6月 (3)

2017年5月 (3)

2017年4月 (1)

2016年12月 (4)

2016年11月 (7)

积分与排名

积分 - 22785

排名 - 15535

阅读排行榜

1. Android 图片加载框架Glide4.0源码完全解析（一）(4752)

2. Android6.0运行时权限管理(4088)

上述中基本数据类型的签名多以大写类型首字母为主，但是引用类型是使用“L”+ 类型路径 + “;”，如String类型则是“Ljava/lang/String;”，数组则是“[” + 类型，如“[I”表示整形数组，如果是Java方法则是“（参数的类型）+ 返回值类型”。

ok，通过GetFieldID获取成员变量ID，如：

```
jfieldID codeErrorID = (*env)->GetFieldID(env, jclass, "codeError", "Ljava/lang/String;");
```

②：获取静态成员变量ID

成员变量分为普通和静态变量，那获取静态变量该如何呢？JNI也为我们封装好了方法：

```
jfieldID (GetStaticFieldID)(JNIEnv, jclass, const char,  
const char);
```

通过GetStaticFieldID函数可以获取到静态变量ID，参数如①一样。举例：

```
jfieldID loginSuccID = (*env)->GetStaticFieldID(env, jclass, "loginSucc", "Ljava/lang/String;");
```

## 通过变量ID获取java成员变量

ok，获取完变量ID，我们就可以通过ID来取得变量了，这里获取成员变量也是分为静态和普通，分别使用：

jobject (GetObjectField)(JNIEnv, jobject, jfieldID);和jobject (GetStaticObjectField)(JNIEnv, jclass, jfieldID);函数。

例如：

```
jstring jcodeError = (*env)->GetObjectField(env, jobject, codeErrorID);  
jstring juserNameError = (*env)->GetObjectField(env, jobject, userNameErrorID);  
jstring jloginSucc = (*env)->GetStaticObjectField(env, jclass, loginSuccID);
```

ok，到这里我们就获取到了Java类中的成员变量，来看下整体代码：

3. Android 网络框架之Retrofit2使用详解及从源码中解析原理(2926)

4. Android 图片加载框架Glide4.0源码完全解析（二）(2448)

5. Android 5.X新特性之为RecyclerView添加下拉刷新和上拉加载及SwipeRefreshLayout实现原理(2195)

### 评论排行榜

1. 2016点滴生活：收获与展望(8)

2. React Native环境配置之Windows版本搭建(1)

3. Android 自定义控件之继承ViewGroup创建新容器(1)

4. Android 设计模式实战之关于封装计费代码库的策略模式详谈(1)

5. Android NDK开发之C调用Java及原生代码断点调试（二）(1)

### 推荐排行榜

```
java x Java2C.c x MainActivity.java x
JNIEXPORT jstring JNICALL Java_com_sanhui_ndkdemo_Java2CJNI_login
(JNIEnv *env, jobject jobj, jstring jUserName, jstring jPSW, jint jcode){
    const char* resultMessage;
    jclass jclazz;
    jobject jobject;
    //1. 加载Java类得到class对象
    // jclass (*FindClass)(JNIEnv*, const char*);
    jclazz = (*env)->FindClass(env, "com/sanhui/ndkdemo/Java2CJNI");
    //2. 获取Java类成员变量
    //2.1 获取Java类成员变量ID
    //jfieldID (*GetFieldID)(JNIEnv*, jclass, const char*, const char*);
    jfieldID codeErrorID = (*env)->GetFieldID(env, jclazz, "codeError", "Ljava/lang/String;");
    jfieldID userNameErrorID = (*env)->GetFieldID(env, jclazz, "userNameError", "Ljava/lang/String;");
    jfieldID loginSuccID = (*env)->GetStaticFieldID(env, jclazz, "loginSucc", "Ljava/lang/String;");
    //2.2 通过成员变量ID获取Java类成员变量
    //jobject (*GetObjectField)(JNIEnv*, jobject, jfieldID);
    jstring jcodeError = (*env)->GetObjectField(env, jobj, codeErrorID);
    jstring juserNameError = (*env)->GetObjectField(env, jobj, userNameErrorID);
    jstring jloginSucc = (*env)->GetStaticObjectField(env, jclazz, loginSuccID);

    return jloginSucc;
}
```

获取到Java中的成员变量，然后返回到Java中，然后通过Toast给打印出来：

```
private void setListener() {
    login.setOnClickListener((v) -> {
        String sUserName = userName.getText().toString().trim();
        String sPSW = psw.getText().toString().trim();
        int sCode = Integer.parseInt(code.getText().toString().trim());
        //可以做些校验过滤，这里就不再做了，而且为了参数类型不同，将验证码转化为int
        String cToJavaResult = jni.login(sUserName, sPSW, sCode);
        Toast.makeText(MainActivity.this, cToJavaResult, Toast.LENGTH_LONG).show();
    });
}
```

来看下执行结果，是否如我们多料想的一样：

1. 2016点滴生活：收获与展望(7)

2. Android NDK开发之C调用Java及原生代码断点调试（二）(3)

3. Android NDK开发之从环境搭建到Demo级十步流(2)

4. Android 网络框架之Retrofit2使用详解及从源码中解析原理(1)

5. Android 设计模式实战之关于封装计费代码库的策略模式详谈(1)

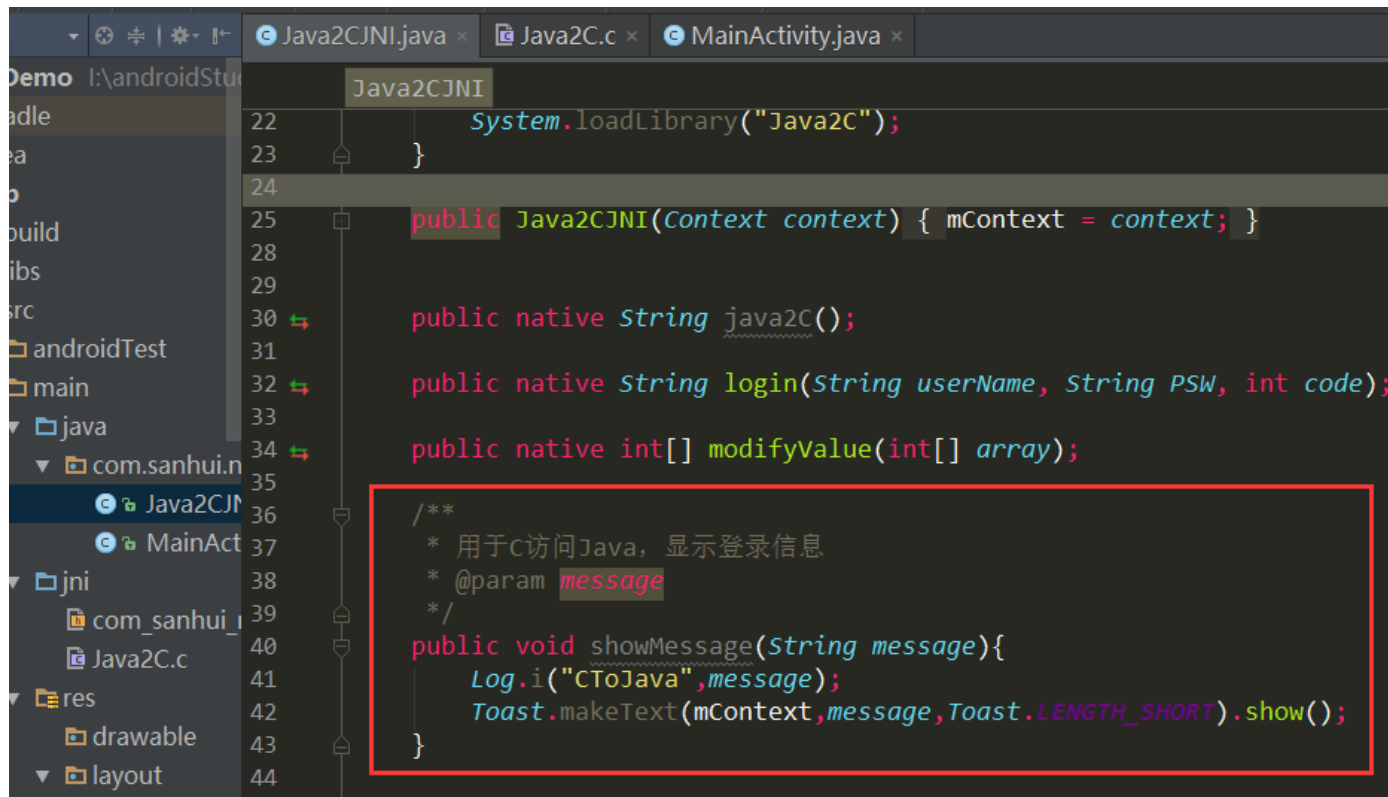


从上图中我们看到的执行结果显然和我们在Java2CJNI中定义的loginSucc成员变量是一样的，由此可以得出结论就是C成功的调用了Java类中的变量。

**注意：**由于获取Java类中的变量需要在原生代码中调用几个方法才能获取到最终的结果，对于性能来说要求的开销过大，所以建议一般不这样直接轰C中调用Java的成员变量，如果有需要建议以参数的形式传递给原生代码。

## C 调用 Java 方法

C调用Java方法和调用成员变量基本是一样的，首先我们现在Java类中定义一个方法，用Toast来显示信息，如：



上面也说过C调用Java方法和变量步骤基本一样，下面来看下基本步骤：

- 1：获取java实例对象的引用
- 2：通过实例对象获取实例方法ID
- 3：通过方法ID调用实际的Java方法

## 获取java实例对象的引用

这一步和C获取变量所介绍的获取方式是一样的，都是通过GetObjectClass或是FindClass函数来获取的，这里就不再赘述，可以参考上面的实力。

## 通过实例对象获取实例方法ID

java中方法分为两类，一类是普通的方法，一类是静态方法。下面来逐一的介绍。

#### ①：获取普通方法ID：

可以通过jmethodID (*GetMethodID*)(*JNIEnv*, jclass, const char, const char);来获取方法ID，这也是JNI已经封装好的原生方法，来解释下这个函数：

**GetMethodID**函数前两个参数就不必多介绍了，其中第三个参数是Java类中的方法名称，对应的是Java2CJNI类中定义的方法：**public void showMessage(String message){}**中的**showMessage**。第四个参数是方法签名，也就是Java类中方法的返回类型，至于什么是签名上面已介绍清楚。

获取方法ID实例：

```
jmethodID showMessage = (*env)->GetMethodID(env, jclass, "showMessage", "(Ljava/lang/String;)V");
```

这里和变量唯一不同的是，方法有可能带参数，那么签名就需要带上参数签名和返回值签名，也就是在（）里的是参数签名，（）外的是返回值签名，如“(Ljava/lang/String;)V”表示是含有一个String类型的参数和一个void的无返回类型。

#### ②：获取静态方法ID：

获取静态方法ID会使用JNI的 jmethodID (*GetStaticMethodID*)(*JNIEnv*, jclass, const char, const char);函数，它的使用和参数与 GetMethodID一样，并没有什么差别。

例如：

```
jmethodID showMessage = (*env)->GetStaticMethodID(env, jclass, "showMessage", "(Ljava/lang/String;)V")
```

## 3：通过方法ID调用实际的Java方法

获取到方法ID后，我们可以通过JNI提供的回调函数来真正的调用Java方法，这里也是分为回调普通方法和静态方法，由于两者基本没什么差别，我们这里就只讲下普通方法的回到。

C回调Java方法会使用Call< type >Method函数来回调实际的方法，例如，我们调用我们显示Toast的无返回值方法：

```
(*env)->CallVoidMethod(env, jobj, showMessage, jloginSucc);
```

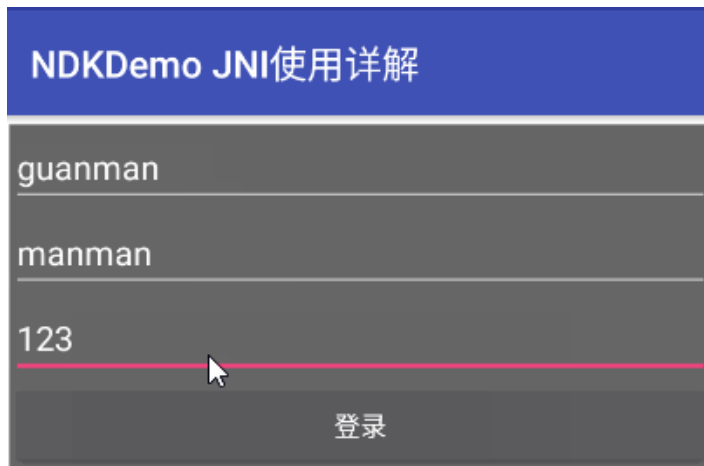
直接的调用CallVoidMethod，它第三个参数传入的是jmethodID类型的方法ID，由之前获取到的，第四个参数是要传递给Java的参数，这里接受的的是一个String累的字符串。

ok，通过上面的三个步骤，我们已调用了Java的方法了，来看下整体的C代码实现吧。



```
va x Java2C.c x MainActivity.java x
//3. 获取Java类方法
//3.1 获取Java类方法ID
//jmethodID (*GetMethodID)(JNIEnv*, jclass, const char*, const char*);
jmethodID showMessage = (*env)->GetMethodID(env, jclass, "showMessage", "(Ljava/lang/String;)V");
if(jcode == 1234){
    const char* cStr;
    jboolean isCopy;
    cStr = (*env)->GetStringUTFChars(env, jUserName, &isCopy);
    int result = strcmp(cStr, "guanmanman");
    if(result == 0){
        //同样的道理psw也可以做一样的验证 TODO
        resultMessage = "success login !";
        //3.2 通过方法ID回调Java方法
        //void (*CallVoidMethod)(JNIEnv*, jobject, jmethodID, ...);
        (*env)->CallVoidMethod(env, jobj, showMessage, jloginSucc);
    }else{
        resultMessage = "error username";
        //3.2 通过方法ID回调Java方法
        //void (*CallVoidMethod)(JNIEnv*, jobject, jmethodID, ...);
        (*env)->CallVoidMethod(env, jobj, showMessage, juserNameError);
    }
}else{
    resultMessage = "error code";
    //3.2 通过方法ID回调Java方法
```

好，来看下执行结果：

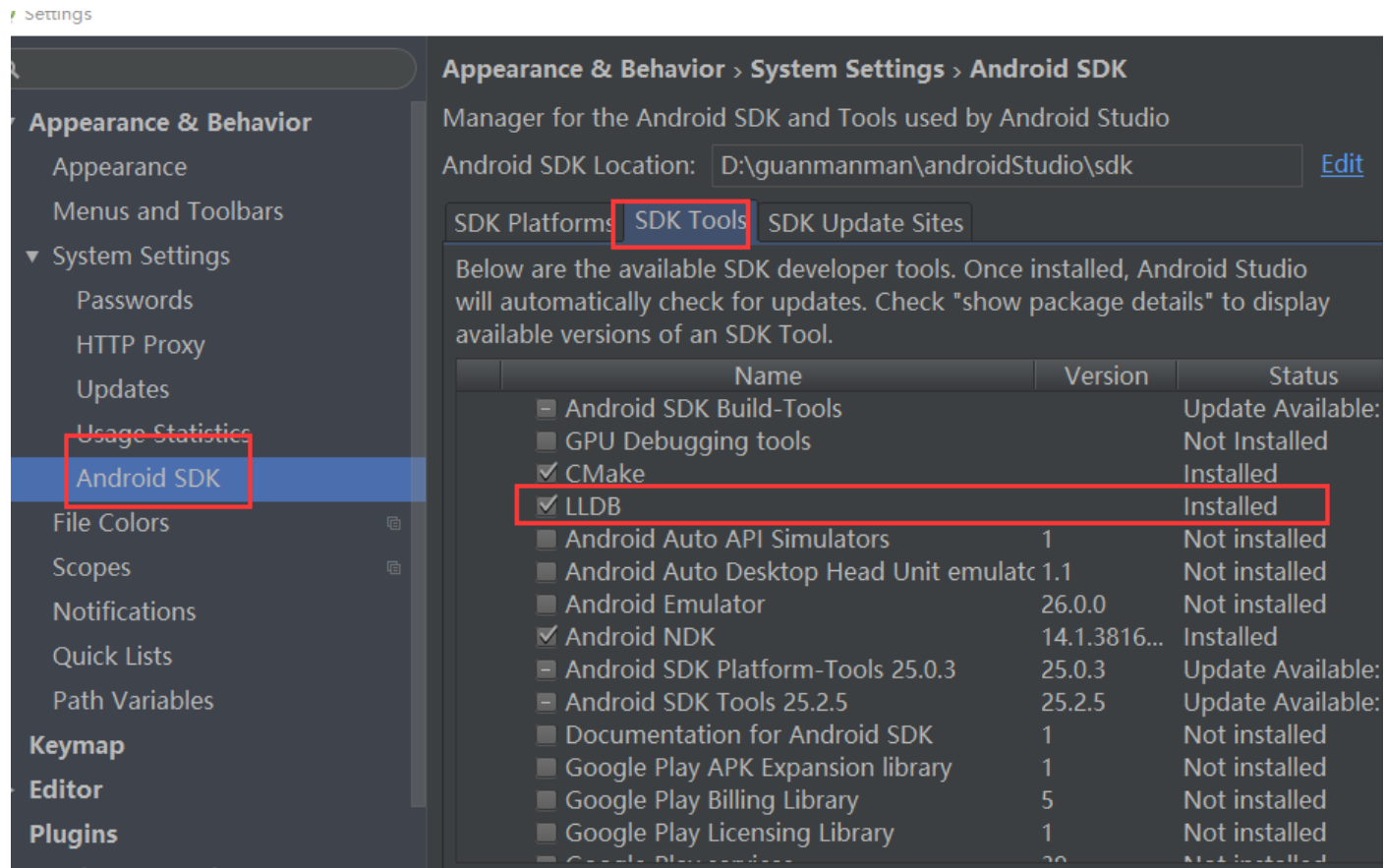


ok，到这里C调用Java就讲完了，下面讲下实用的C原生代码怎么断点调试。

## Androidstudio中原生代码断点调试

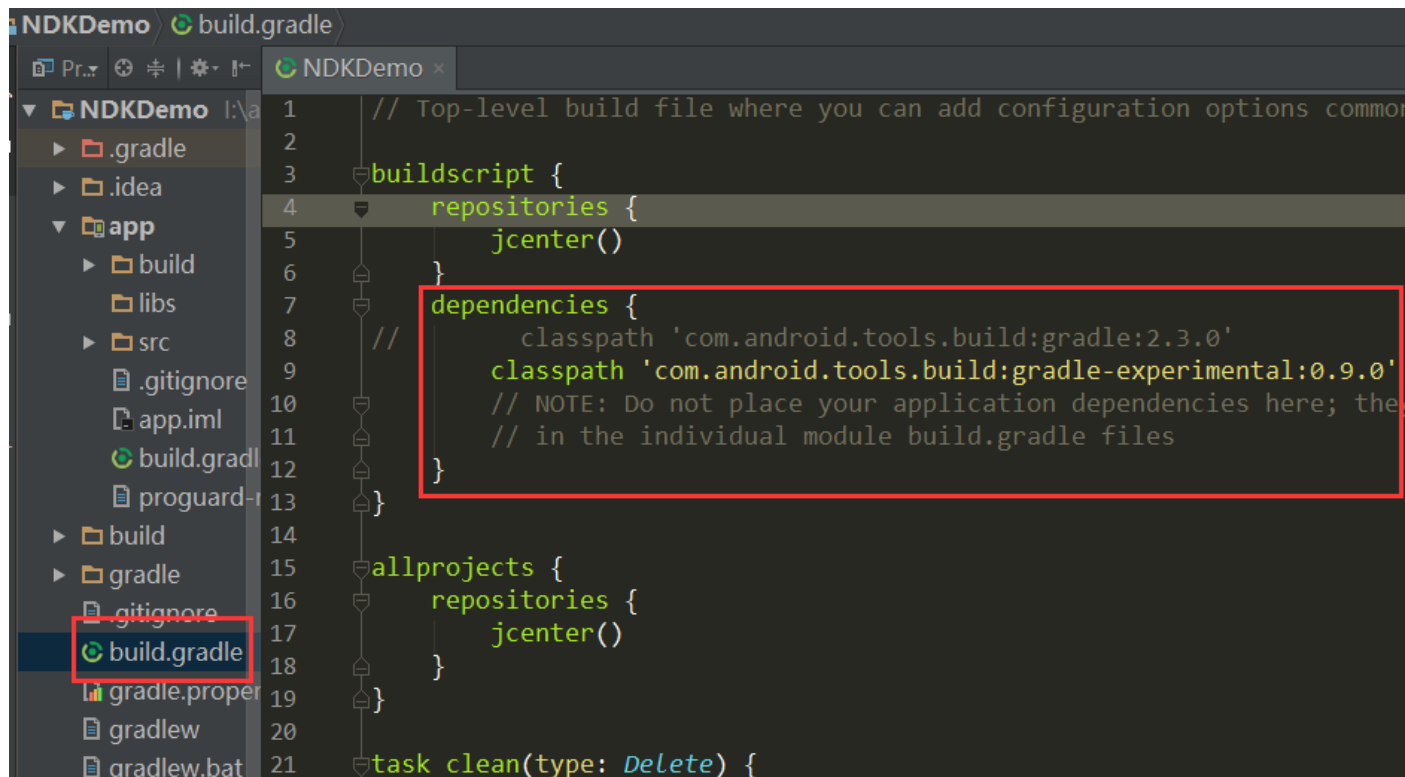
### 下载C/C++调试器LLDB

在androidstudio->File->settings->androidSDK->SDK Tools中下载LLDB:



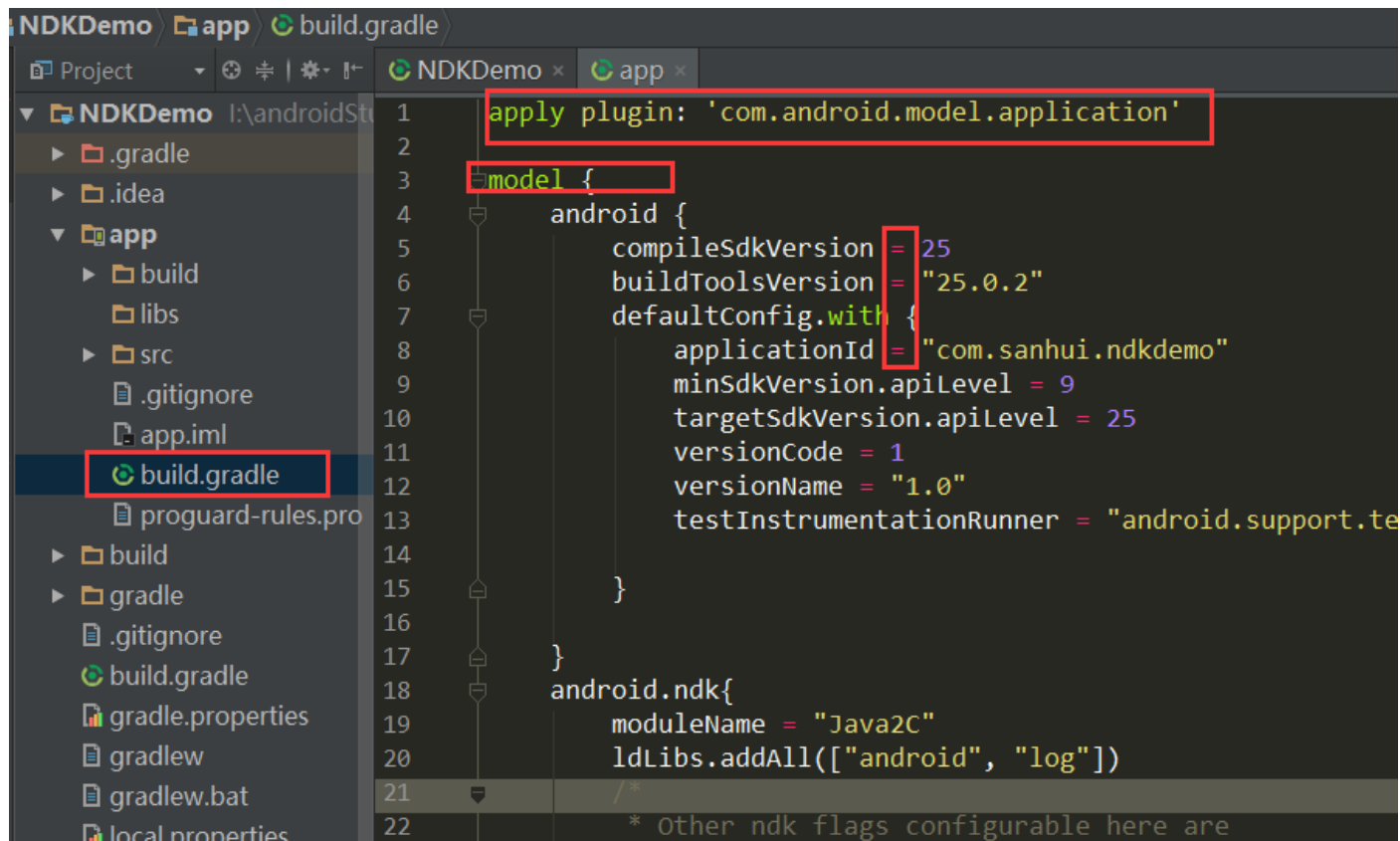
## 在项目目录下的build.gradle文件添加对gradle-experimental的依赖

用项目对gradle-experimental的依赖代替原本项目对gradle的依赖。



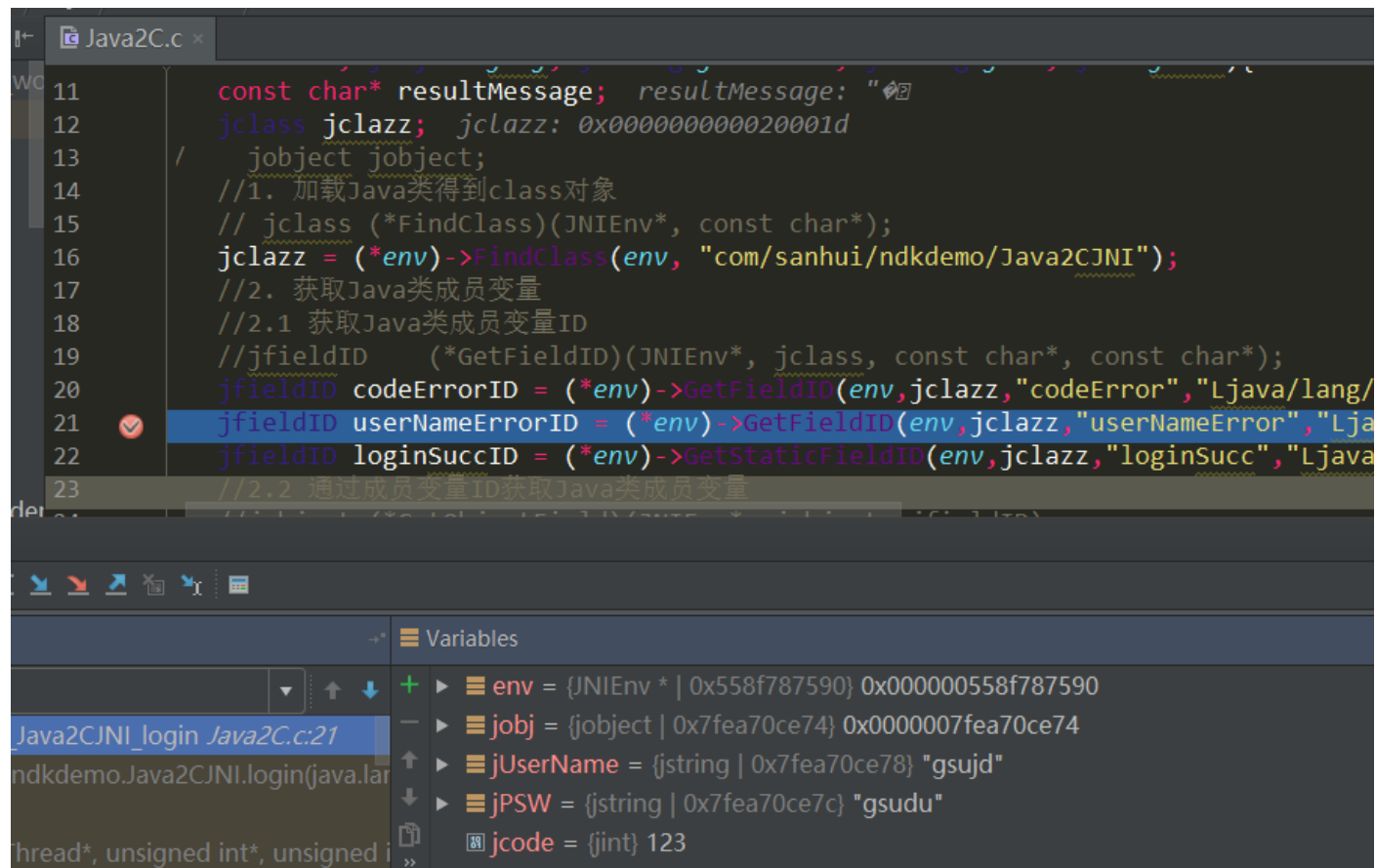
## 配置工程下的build.gradle

- ① 使用com.android.model.application替代原来的com.android.application
- ② 把原有的配置存放在model{}中
- ③ 所有的配置属性使用等号(=)连接



## DUG运行

原生C代码中断点，然后执行dug运行模式。



The screenshot shows an IDE with a C file named `Java2C.c`. The code is implementing JNI functions to call Java. Line 21, `jfieldID userNameErrorID = (*env)->GetFieldID(env, jclass, "userNameError", "Ljava/lang/`, is highlighted with a blue bar and a red checkmark icon. Below the code, the `Variables` window is open, showing the following variables:

- `env` = {JNIEnv \* | 0x558f787590} 0x000000558f787590
- `obj` = {jobject | 0x7fea70ce74} 0x0000007fea70ce74
- `jUserName` = {jstring | 0x7fea70ce78} "gsujd"
- `jPSW` = {jstring | 0x7fea70ce7c} "gsudu"
- `jcode` = {jint} 123

ok，接下来就可以执行你的源代码了。

好了，今天就讲到这里吧。

请关注微信公众号。谢谢

[好文要顶](#)[关注我](#)[收藏该文](#)**管满满**

关注 - 4

粉丝 - 13

[+加关注](#)

3

0

[« 上一篇：Android NDK开发之从Java与C互调中详解JNI使用（一）](#)[» 下一篇：Android 图片加载框架Picasso基本使用和源码完全解析（巨细无比）](#)

posted @ 2017-05-11 11:52 管满满 阅读(1157) 评论(1) 编辑 收藏

#### 评论列表

#1楼 2017-05-11 15:47 fgh545

参考资料：[www.hldypcb.com](http://www.hldypcb.com)

支持(0) 反对(0)

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

【推荐】50万行VC++源码: 大型组态工控、电力仿真CAD与GIS源码库

【推荐】腾讯云免费实验室，1小时搭建人工智能应用

【新闻】H3 BPM体验平台全面上线



#### 最新IT新闻:

- 铁路12306 App迎3.0版更新：全新界面
  - 若福克斯资产出售 可能会让Hulu成Netflix强大对手
  - 三星260亿美元的豪赌：想垄断DRAM和NAND闪存市场
  - 苹果还是没有放弃？屏下指纹识别专利被曝光
  - Google员工和Google.org在2017年向非营利组织捐赠了2.6亿美元
- » 更多新闻...





**最新知识库文章:**

- 以操作系统的角度述说线程与进程
- 软件测试转型之路
- 门内门外看招聘
- 大道至简，职场上做人做事做管理
- 关于编程，你的练习是不是有效的？
- » 更多知识库文章...

Copyright ©2017 管满满