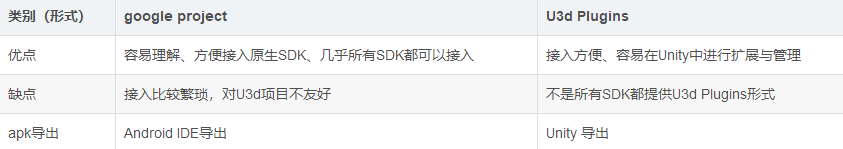
<https://blog.csdn.net/yang8456211/article/details/51356193>

Android SDK的接入一般分为两种：

1. 一种是把Unity的工程导出google project的形式进行接入
2. 另一种是通过把Android的工程做成Plugins的形式进行接入



建议：

如果做一个sdk，还是推荐分开Android形式的SDK（可以是Library），与Unity3d形式的SDK（Plugins），因为一般项目会有自己的SDK架构，而让他们打破这个架构，导出Google project的形式进行接入，这无疑是很难被接受的。

只有当需要在Activity的生命周期中执行一些操作时，我们才需要一个中间Activity去完成这些与生命周期相关的操作，而其他情况下，一个Class足够了。

下面一个例子采用两种方式接入sdk

**（1）创建android sdk**

**public** **class** AndroidSDK {

**private** **static** Context *context*;

**private** **static** Activity *activity*;

**private** **static** AlertDialog.Builder *alertDialog*;

**public** **static** **void** init(Context context) {

context = context.getApplicationContext();

*activity* = (Activity) context;

}

**public** **static** **int** add(**int** arg1, **int** arg2) {

**return** arg1 + arg2;

}

**public** **static** **void** showMessage(**final** String msg) {

*activity*.runOnUiThread(**new** Runnable() {

**public** **void** run() {

Toast.*makeText*(*context*, msg, Toast.*LENGTH\_LONG*).show();

}

});

}

**public** **static** **void** showAlertDialog() {

*alertDialog* = **new** AlertDialog.Builder(*activity*).setTitle("弹出窗口").setMessage(*context*.getResources().getIdentifier("msgAlert", "string", *context*.getPackageName())).setPositiveButton("确认", **new** DialogInterface.OnClickListener() {

@Override

**public** **void** onClick(DialogInterface arg0, **int** arg1) {

*activity*.finish();

}

});

*activity*.runOnUiThread(**new** Runnable() {

**public** **void** run() {

*alertDialog*.show();

}

});

}

}

几个问题：

1. 不使用Activity，那Activity上下文怎么来

使用init方法让Unity传当前上下文进来，我们就能拿到当前的Activity和应用上下文

1. 怎么弹出窗口？怎么对界面进行操作？

Unity调用Android方法默认不是在UI主线程上执行的，所以如果想对UI界面进行操作，那就要使用runOnUiThread才行

1. 使用Plugins接入时候，怎么才能读到R的资源

这个问题困扰了我挺久的，因为直接把libs放入到Plugins中会导致读取不到资源，最后是通过反编译各种第三方SDK，才找到方法。此处是通过unityContext.getResources().getIdentifier(“msgAlert”, “string”, unityContext.getPackageName())这种java反射机制获取。（这里是获取了一个string,类比其他）

1. 怎么发布我们的sdk呢？

此时我们已经写好了sdk，这个sdk比较健壮，怎么发布都行，可以勾上isLibrary以Android的库文件形式进行发布；也可以分离出jar和res，以代码和资源的方式进行发布

Unity的c#代码：

public class CallAndroid : MonoBehaviour {

AndroidJavaClass javaClass;

AndroidJavaClass androidCallClass;

AndroidJavaObject currentActivity;

void Start () {

javaClass = new AndroidJavaClass("com.unity3d.player.UnityPlayer");

currentActivity = javaClass.GetStatic<AndroidJavaObject>("currentActivity");

androidCallClass = new AndroidJavaClass("com.example.androidsdk.AndroidSDK");

androidCallClass.CallStatic("init", currentActivity);

}

void OnGUI()

{

GUI.skin.textArea.fontSize = 50;

GUI.skin.button.fontSize = 50;

if (GUI.Button(new Rect(100, 300, 450, 300), "add"))

{

int sum = androidCallClass.CallStatic<int>("add", 1, 2);

Debug.Log("sum is " + sum);

}

if (GUI.Button(new Rect(600, 300, 450, 300), "showMessage"))

{

androidCallClass.CallStatic("showMessage", "显示这段文字");

}

if (GUI.Button(new Rect(100, 700, 450, 300), "showAlertDialog"))

{

androidCallClass.CallStatic("showAlertDialog");

}

}

}

**方式一：导出google android工程形式**

将CallAndroidSDK.jar包放入工程的额libs中，将stringnew.xml放入res/values中，其中stringnew.xml文件内容如下：

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<resources>

<string name=*"msgAlert"*>Unity</string>

</resources>

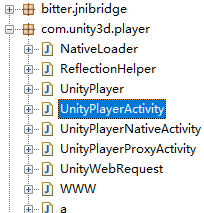
**Libs目录**

Libs有三部分：前两者（armeabi-v7a、x86）表示手机的两种硬件设备，包含了各自所需的so文件，我们可以在Unity的Player Setting->Other Setting->Device Filter中来进行选择。而unity\_classes.jar就是UnityPlayerActivity类所在的jar包，其实就是Unity目录下的Editor\Data\PlaybackEngines\AndroidPlayer\Variations\il2cpp\Development\Classes\classes.jar



**Classes.jar**

反编译结果：

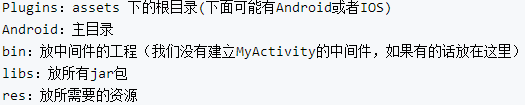


已经验证该方式可以正常运行

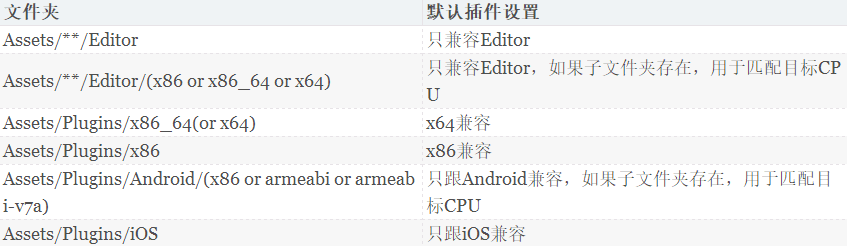
**方式二：以Plugins方式**

这种方式更简单，简而言之就是以unity要求的形式放好你的资源和jar包，然后就能够运行了。

在Assets中添加一个Plugins文件夹，然后在Plugins中添加Android文件夹



**如果Plugins文件夹中存在”Plugins/x86”和”Plugins/x86\_64”这两个文件夹，则dll必须放入这两个文件夹中，否则会出现找不到dll的情况**



**Plugins/Android**

我们可以通过Plugins/Android中添加东西来影响Unity生成的Android项目

AndroidManifest.xml

我们可以直接在Android目录下添加一个AndroidManifest.xml文件，这个文件会影响最终的Android项目清单文件。注意，这里的程度只是影响，而不是完全替换，比如package会最终替换为Bundle Identifier的设置，如果没有以下几行：

<uses-feature android:name="android.hardware.touchscreen" android:required="false" />

<uses-feature android:name="android.hardware.touchscreen.multitouch" android:required="false" />

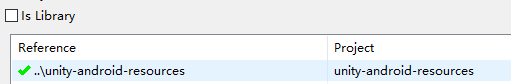
<uses-feature android:name="android.hardware.touchscreen.multitouch.distinct" android:required="false" />

如果是默认设置它也会加上去（已经验证）

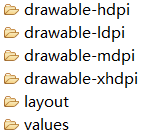
PS：AndroidManifest.xml要放在Android目录下才会对主清单文件起作用，如果你放到其他目录，比如test，则不会对最终的Android项目有作用。

Res目录

可以在Android目录下增加一个res目录。不管这个目录是空的还是有东西，导出来的Android项目都会多一个叫unity-android-resources的Libarary工程，而我们的主Android工程则引用了这个Library，如图：



观察这个Library就知道，这个工程主要就是存放资源的，比如我们这里把东西放到res目录中，则这些多出来的东西就会放到Library工程中。注意，res目录的东西不能随便乱放，要按照合格的目录结构来，一般如下：



比如我们添加一个test文件在res中，会发现出现下面的错误提示：



Assets目录

Assets目录和res目录类似，都是存放资源的，只是res目录里的资源会在R文件中生成ID，而asset目录不会生成ID，需要自己手动根据路径来访问资源

Jar包

网上相关资源，多数都说要把jar包放在Plugins/Android目录下的libs或bin目录，有可能是unity5做了修改，发现jar包不管放到哪，只要在Assets目录下，就能起作用，成为最终Android工程的libs中的内容

Library工程

你可以在Plugins/Android目录下放置一个Android Library工程，只要设置得当，这个工程就会被Unity识别并在最终生成的Android工程中引用。这个也是接入Android SDK时主要使用的方式，十分方便并且易于管理（毕竟这让你知道那些资源、jar包是归属于那个SDK的。而如果所有的资源都放在res目录中，jar包都放在libs目录，这样就不知道某个SDK原本的东西在哪了）

