How to Use Android API

in UE4 Project

Index

[1) Write an xml script to call functions compiled in Android SDK APIs or in a JAR file 3](#_Toc2953748)

[2) Register XML file to integrate to the project 5](#_Toc2953749)

[3) Use your own C++ class to call function defined in the xml file 6](#_Toc2953750)

[Usage of FJavaWrapper::FindMethod（） 7](#_Toc2953751)

[Usage of FJavaWrapper::CallBooleanMethod（） 8](#_Toc2953752)

**This document mainly introduces the approach that how to call Android SDK APIs or the methods defined in a JAR file in UE4 project.**

# Write an xml script to call functions compiled in Android SDK APIs or in a JAR file

1. Create an xml file under \*ProjectName*\Source\ProjectName\, under the same folder as ProjectName.build.cs
2. Write the code referring to the demo code below. You need to insert your Android code accordingly.
3. If you are using JAR file, you need to write this tag

<resourceCopies></resourceCopies>

And indicate path of your JAR file here.

1. Save this xml file to \Source\ProjectName\filename.xml

Please refer to the following code:

1. **<root** xmlns:android="http://schemas.android.com/apk/res/android"**>**
2. **<init>**
3. <!—Write your initialization contents.--**>**
4. <!—Eample--**>**
5. **<log** text="Pico Test for Yeti"**/>**
6. **</init>**
8. **<resourceCopies>**
9. <!—Indicate path of 3rd-party library you need to include(.so file/.jar file) --**>**
10. <!—Example--**>**
11. **<copyFile** src="$S(PluginDir)/lib/javalib.jar" dst="$S(BuildDir)/libs/javalib.jar" **/>**
12. **</resourceCopies>**
14. **<gameActivityImportAdditions>**
15. **<insert>**
16. <!—Indicate the packages you need to include.--**>**
17. <!—Example--**>**
18. import android.widget.Toast;
19. import android.hardware.Camera;
20. import android.hardware.Camera.CameraInfo;
21. import android.hardware.Camera.Parameters;
22. import android.hardware.Camera.PreviewCallback;
23. import android.graphics.SurfaceTexture;
24. import android.graphics.ImageFormat;
25. import android.graphics.PixelFormat;
26. import java.util.List;
27. import java.io.IOException;
28. import android.util.Log;
29. import com.pvr.PvrManager;
30. **</insert>**
31. **</gameActivityImportAdditions>**
33. **<gameActivityClassAdditions>**
34. **<insert>**
35. <!—Register the API fuction you need to call--**>**
36. <!—Example--**>**
37. private static PvrManager mPvrManager = null;
38. public boolean AndroidThunkJava\_UEPvr\_SetLED(int led, int color, int blink, int ontime, int offtime)
39. {
40. boolean ret = false;
41. if(null != mPvrManager)
42. {
43. mPvrManager.setLedStatus(led, color, blink, ontime, offtime);
44. ret = true;
45. Log.debug("LED brightness: " + (ret == true?"succeed":"failed"));
46. }
47. return ret;
48. }
49. }
50. **</insert>**
51. **</gameActivityClassAdditions>**
53. **<gameActivityOnCreateAdditions>**
54. **<insert>**
55. <!—Write your optional additions to GameActivity oncreate in GameActivity.java--**>**
56. **</insert>**
57. **</gameActivityOnCreateAdditions>**
58. **</root>**

## Register XML file to integrate to the project

Then you need to insert build code into *ProjectName*.build.cs to compile this project.

1. Copy the following code.

if（Target.Platform == UnrealTargetPlatform.Android）

{

PrivateDependencyModuleNames.AddRange(new string[] { "Launch" });

String LocalPath = Utils.MakePathRelativeTo(ModuleDirectory, BuildConfiguration.RelativeEnginePath);

AdditionalPropertiesForReceipt.Add(new ReceiptProperty("AndroidPlugin", (LocalPath + "/ filename.xml")));

}

1. Paste this segment into *ProjectName*.build.cs, (Highlighted in red.). Be aware that “filename.xml” should be
2. **using** UnrealBuildTool;
3. **using** System.IO;
4. **public** **class** MYJar : ModuleRules
5. {
6. **public** MYJar(ReadOnlyTargetRules Target) : base(Target)
7. {
8. PCHUsage = PCHUsageMode.UseExplicitOrSharedPCHs;
9. PublicDependencyModuleNames.AddRange(**new** string[] { "Core", "CoreUObject", "Engine", "InputCore" });
10. PrivateDependencyModuleNames.AddRange(**new** string[] {  });
11. // Uncomment if you are using Slate UI
12. // PrivateDependencyModuleNames.AddRange(new string[] { "Slate", "SlateCore" });
14. // Uncomment if you are using online features
15. // PrivateDependencyModuleNames.Add("OnlineSubsystem");
17. // To include OnlineSubsystemSteam, add it to the plugins section in your uproject file with the Enabled attribute set to true
18. **if** (Target.Platform == UnrealTargetPlatform.Android)
19. {
20. PrivateDependencyModuleNames.AddRange(**new** string[] { "Launch" });
21. AdditionalPropertiesForReceipt.Add(**new** ReceiptProperty("AndroidPlugin", ModuleDirectory + "/CPPJAVAS.xml"));
22. }
23. }
24. }

## Use your own C++ class to call function defined in the xml file

Please follow these steps to call functions defined in xml file.

1. Ensure “AndroidJNI.h” and “AndroidApplication.h”to be included in the cpp file.
2. Get Java runtime environment by using FAndroidApplication::GetJavaEnv().
3. Use JavaWrapper::FindMethod() to obtain ID of the java function defined in xml file.
4. For example, use FJavaWrapper::CallBooleanMethod() to invoke Android API which returns boolean value. For other different definitions, you need use corresponding FJavaWrapper methods.

Please refer to the following code:

1. // Fill out your copyright notice in the Description page of Project Settings.
3. #include "MyBlueprintJarTest.h"
4. #if PLATFORM\_ANDROID
5. #include "Android/AndroidApplication.h"
6. #include "Android/AndroidJNI.h"
7. #include <android/log.h>
8. #endif
9. **void** UMyBlueprintJarTest::TestLED(**int** led, **int** color, **int** blink, **int** ontime, **int** offtime)
10. {
11. #if PLATFORM\_ANDROID
12. **if** (JNIEnv\* Env = FAndroidApplication::GetJavaEnv())
13. {
14. **static** jmethodID Method = FJavaWrapper::FindMethod(Env, FJavaWrapper::GameActivityClassID, "AndroidThunkJava\_UEPvr\_SetLED", "(IIIII)Z", **false**);
15. FJavaWrapper::CallBooleanMethod(Env, FJavaWrapper::GameActivityThis, Method, led, color, blink, ontime, offtime);
16. }
17. #endif
18. }

## Usage of FJavaWrapper::FindMethod（）

Call Example:

FJavaWrapper::FindMethod(Env, FJavaWrapper::GameActivityClassID, " AndroidThunkJava\_UEPvr\_SetLED ", "(IIIII)Z", false);

Parameter:

1. Env:

Java environment, you can get it using FAndroidApplication::GetJavaEnv() method，

1. FJavaWrapper::GameActivityClassID

This parameter is the java function you added in GameActivity of the xml file. GameAcitivity. You don’t have to modify it. This parameter indicate the function you need is declared in this xml file.

1. AndroidThunkJava\_UEPvr\_SetCurrentBrightness

This is the function name you defined in xml file. You need to modify it according to xml file you wrote.

1. (IIIII)Z

This parameter is the input parameter and return parameter of the call function. Inside the bracket are the input type. And outside the bracket are the returned type. In this example, it means input parameters are 5 int type, and return type is 1 bool type.

Please refer to the following list for the type.

|  |  |
| --- | --- |
| Data Type | Symbol |
| Bool | Z |
| Int | I |
| Double | D |
| FString | Ljava/lang/String |
| Void | V |
| int[] | [I |
| double[] | [D |

## Usage of FJavaWrapper::CallBooleanMethod（）

Call Example:

FJavaWrapper::CallBooleanMethod(Env, FJavaWrapper::GameActivityThis, Method, index,level)

Parameters:

1. Env

Java environment, you can get it using FAndroidApplication::GetJavaEnv() method.

1. FJavaWrapper::GameActivityThis

This indicate the GameActivity. You don’t need to modify it.

1. Method

This is the method ID you need to call. In the example, it is the returned value of FJavaWrapper::FinMethod()

1. led, color, blink, ontime, offtime

These are input parameters of the function your defined in xml file. In the example, these are necessary parameters to control LED. You need to modify it according to xml file you wrote.