How to use JAR file in Unity project on Pico device

Index

[Method 1: Inheriting from Class MainActivity 3](#_Toc534994021)

[1. Writing JAR file in Android Studio 3](#_Toc534994022)

[2. Use the JAR file in Unity project 5](#_Toc534994023)

[Method 2: Non-inheriting the main Activity 7](#_Toc534994024)

[1. Writing JAR file in Android Studio 7](#_Toc534994025)

[2. Use the JAR file in Unity project 8](#_Toc534994026)

[Method 3: Unity methods be invoked passively by JAR file using UnitySendMessage 9](#_Toc534994027)

[1. Writing JAR file 9](#_Toc534994028)

[2. Define the methods to be invoked in Unity project 10](#_Toc534994029)

[Notes 11](#_Toc534994030)

**This document mainly introduces three approaches that how to write and call methods defined in a JAR file, which allows developer to call Android methods in Unity project on Pico device.**

* Using Method 1, the name of the main activity of Unity application need to be changed, which is capable to do some initialization and customization work in activity lifecycle.
* Using Method 2, Unity application can remain its original main activity and simply import multiple JAR files.
* Using Method 3, Unity methods will be invoked automatically by JAR files. In this way, Unity application doesn’t need to call the methods in JAR file by itself consecutively.

# Method 1: Inheriting from Class MainActivity

## 1. Writing JAR file in Android Studio

1. Import vractivity.jar in SDK assets/plugin/android path in app->libs, and right click vractivity.jar to select “add as library”
2. Inherit from UnityPlayerNativeActivityPico class, add

*public static Activity unityActivity = null;* add *unityActivity = this;* in method onCreate(),

1. Comment out the line setContentView();



1. Add the following code on the app-> build.gradle root node,

task createJarDebug(type: Jar,dependsOn: ['build']) {  
 baseName "test"  
 from 'build/intermediates/javac/debug/compileDebugJavaWithJavac/classes'  
 include('\*\*')  
 exclude('\*\*/R.class')  
 exclude('\*\*/R$\*.class')

exclude('\*\*/BuildConfig.class')  
}

//add the following code if you want to obfuscate code

task proguard(type: proguard.gradle.ProGuardTask, dependsOn: buildJar) {  
// input path  
 injars "build/libs/test.jar"  
// output path  
 outjars "lib/testProguard.jar"  
// add config info  
 configuration 'proguard-rules.pro'  
}

1. Enter *gradlew createJarDebug* in Android Studio Terminal

You need to generate the conflation package by typing

*gradlew createJarDebug proguard*

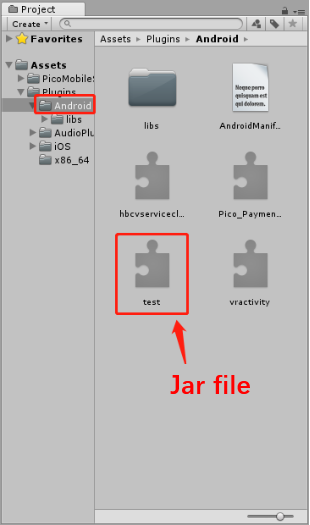
If prompted build success, the package succeeds

JAR file path: app/build/libs/test.jar

Provide the JAR file info (Package name, Class name and Interface name) to Unity engineer.

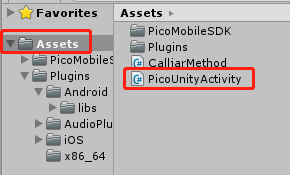
## 2. Use the JAR file in Unity project

1) Copy the JAR file to the Assets/Plugins/Android/ folder



1. Copy the PicoUnityActivity.cs to the Assets folder.





1. Modify the className property of the *CurrentActivity()* method in PicoUnityActivity.cs to the Class name of the JAR
2. **public** **class** PicoUnityActivity
3. {
4. **private** **static** AndroidJavaObject picoUnityAcvity = **null**;
5. **private** **static** AndroidJavaObject CurrentActivity ()
6. {
7. **if** (Application.platform == RuntimePlatform.Android) {
8. string className= "com.picovr.openrecenter.MainActivity";
10. **if** (picoUnityAcvity == **null**) {
11. picoUnityAcvity = **new** AndroidJavaClass (className).GetStatic<AndroidJavaObject> ("unityActivity");
12. }
13. }
15. **return** picoUnityAcvity;
16. }
17. In the script file, Use “CallObjectMethod” interface to call the methods defined in JAR files as below.
18. **public** **class** CalljarMethod : MonoBehaviour {
20. // Use this for initialization
21. **void** Start () {
22. PicoUnityActivity.CallObjectMethod("startJar")
23. }
25. // Update is called once per frame
26. **void** Update(){
28. }

# Method 2: Non-inheriting the main Activity

## 1. Writing JAR file in Android Studio

1. Create a new class and write your own method (here's how to create the file and update the media library

*package com.pico.jartest*

*Import …*

*public class JarClass {*

*public void updateFile(Context context, String fullFileName) {*

*Log.e(TAG, "updateFile: ");*

*Intent intent = new Intent(Intent.ACTION\_MEDIA\_SCANNER\_SCAN\_FILE);*

*intent.setData(Uri.fromFile(new File(fullFileName)));*

*context.sendBroadcast(intent);*

*}}*

1. Add the following code on the app-> build.gradle root node, which is the same level as android and dependencies

task createJarDebug(type: Jar,dependsOn: ['build']) {  
 baseName "test"  
 from 'build/intermediates/javac/debug/compileDebugJavaWithJavac/classes'   
 include('\*\*')  
 exclude('\*\*/R.class')  
 exclude('\*\*/R$\*.class')  
 exclude('\*\*/BuildConfig.class')  
 exclude('\*\*/MainActivity.class')  
 exclude('\*\*/MainActivity$\*.class')  
}

1. In Terminal, type gradlew createJarDebug (you need to generate the obguile package, type gradlew createJarDebug proguard)

If prompted build success, the package succeeds

Jar package path: app/build/libs/test.jar

Copy to Unity engineer and tell them package name. Class name, interface name

## 2. Use the JAR file in Unity project

1. Copy the Jar package to the Assets/Plugins/Android/ directory
2. Create a new script, use “PackageName.ClassName” to initialize object of the class defined in JAR file.

*public class filetest : MonoBehaviour {*

*string fullFileName;*

*AndroidJavaObject updateFileManager;*

*AndroidJavaObject ActivityContext;*

*private void Start()*

*{*

*fullFileName = "/storage/emulated/0/Download/update.txt";*

*updateFileManager = new AndroidJavaObject("com.picovr.updateanyfile.UpdateManager");*

*ActivityContext = new AndroidJavaClass("com.unity3d.player.UnityPlayer").GetStatic<AndroidJavaObject>("currentActivity");*

*}*

*public void updateFile()*

*{*

***updateFileManager.Call("updateFile", ActivityContext, fullFileName);***

*}*

*}*

# Method 3: Unity methods be invoked passively by JAR file using UnitySendMessage

## Writing JAR file

1. import classes.jar to app -> libs, right click it and choose *Add As Library*

path of classes.jar :

*[your unity intall path]\Editor\Data\PlaybackEngines*

*\AndroidPlayer\Variations\mono\Release\Classes*

1. inherit UnityPlayerActivity class
2. call UnityPlayer.UnitySendMessage(String objectName, String methodName, String methodParameter) to invoke Unity methods.

eg: send Bluetooth state change state once u receive the broadcast.

*… …*

*switch (message) {*

*case BluetoothAdapter.STATE\_OFF:*

*UnityPlayer.UnitySendMessage("SetState", "setBluetoothState", "bt\_state\_off");*

*break;*

*… …*

1. Comment out the line *setContentView(R.layout.activity\_main);*
2. Add the following code in the app-> build.gradle root node (which is the same level as android and dependencies).

task createJarDebug(type: Jar,dependsOn: ['build']) {  
 baseName "test"  
 from 'build/intermediates/javac/debug/compileDebugJavaWithJavac/classe s'  
 include('\*\*')  
 exclude('\*\*/R.class')  
 exclude('\*\*/R$\*.class')

exclude('\*\*/BuildConfig.class')  
}

//add the following code if you want to obfuscate code

task proguard(type: proguard.gradle.ProGuardTask, dependsOn: buildJar) {  
// input path  
 injars "build/libs/test.jar"  
// output path  
 outjars "lib/testProguard.jar"  
// add config info  
 configuration 'proguard-rules.pro'  
}

1. In Android Terminal, type *gradlew createJarDebug*

You need to generate the conflation package by typing *gradlew createJarDebug proguard*

If prompted build success, the package succeeds

Jar package path: app/build/libs/test.jar

Copy to Unity engineer and tell them package name. Class name, interface name

## 2. Define the methods to be invoked in Unity project

1) Copy the JAR file to Assets/Plugins/Android/ path

2) To be invoked by UnitySendMessage(String objectName, String method same, String methodParameter) method in the JAR file, In Unity project please create a GameObject by name of objectName, and add a script that include the method by name of methodName, which has the same type of the input parameter as methodParameter. Please refer to the example codes below:

*public class test : MonoBehaviour {*

*public Text wifiText;*

*public void setWifiState(string s)*

*{*

*wifiText.text = s;*

*}*

*}*

# Notes

1. If there is a dependency of the JAR file, please put all the related JAR files to the Unity project as well.

2. It is required to register the relevant Android components in Unity project AndroidManifest.xml script.