

# Unity Export 导出到 Android Studio 再打包大致过程

deepwaterooo

November 23, 2022







## Contents

<b>1 导出的 unity 项目文件大致是这样的</b>	<b>1</b>
<b>2 Android 创建、unity 导入</b>	<b>2</b>
2.1 首先新建一个 Android 项目	2
2.2 将 unity 项目以 Module 的方式导入 Android	2
2.3 选择 unityLibrary 导入。点击 Finish	3
2.4 导入之后，为 Android 添加 unityLibrary 的引用	3
2.5 配置 Android 以及 unity 的 build.gradle 文件	3
<b>3 Android 启动运行 unity</b>	<b>4</b>
3.1 在 unity 的 AndroidManifest.xml 文件	4
3.2 在 app 的 AndroidManifest.xml 文件里，在图中位置加入这两行代码：	4
3.3 在 app 的 build.gradle 里加入这行代码。	5
3.4 在 app 的 main->res->values->strings.xml 里加入这行代码	5
3.5 点击按钮启动 unity(画蛇添足)	7
<b>4 启动运行</b>	<b>8</b>
<b>5 那么现在就是说：安卓 SDK 与 unity 的交互与打包基本没有问题了</b>	<b>8</b>

## 1 导出的 unity 项目文件大致是这样的

- 大致过程记一下, 用作参考, 原理还没有吃透, 细节又比较多, 容易忘记. 作个笔记记一下, 给自己用作参考

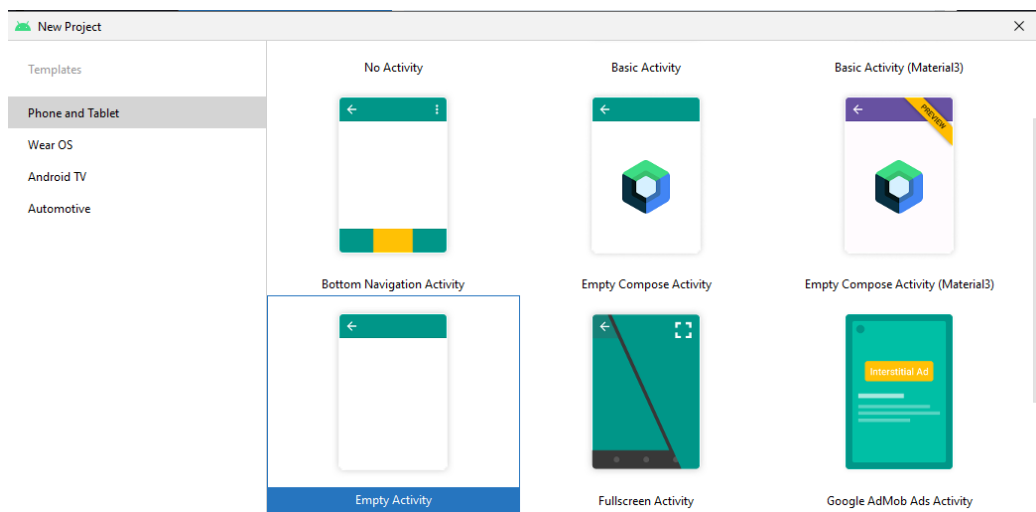
F (F:) > tmp > prevTetris19 >

Name	Date modified
 launcher	11/23/2022 1
 unityLibrary	11/23/2022 1
 build.gradle	11/23/2022 1
 gradle.properties	11/23/2022 1
 local.properties	11/23/2022 1
 settings.gradle	11/23/2022 1

- 下面是 2019 年的版本可以打出两个文件夹, 一个主工程, 一个类库的导出包, 2017 年我用的版本打不出来, 还需要想得再深一点多点儿, 到可以按照这个笔记过程打包才行

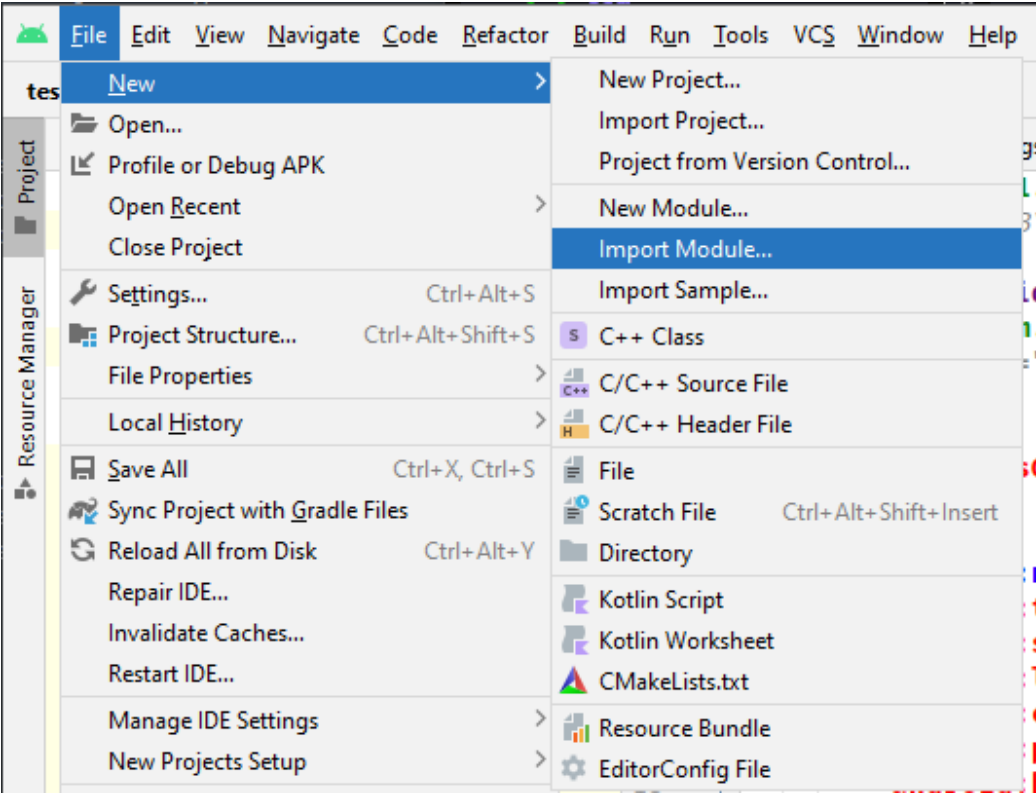
## 2 Android 创建、unity 导入

### 2.1 首先新建一个 Android 项目

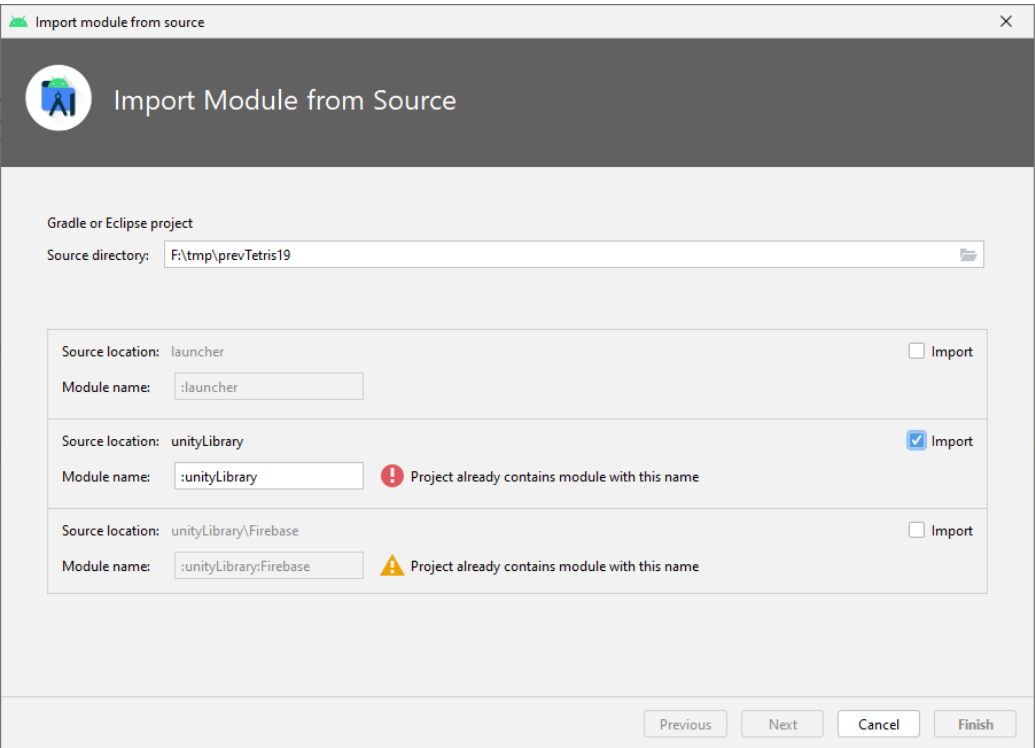


- 包名 Package name 跟 unity 的包名设置成一致, unity 包名一般是 **com.unity3d.player**。包名不一致的话, 我试过也可以实现, 但是在调用的时候要指明包, 容易混淆, 可能还有其他的一些问题, 个人也不是很清楚。推荐保持一致, 避免麻烦。Android 项目名 Name 等随意。

2.2 将 unity 项目以 Module 的方式导入 Android

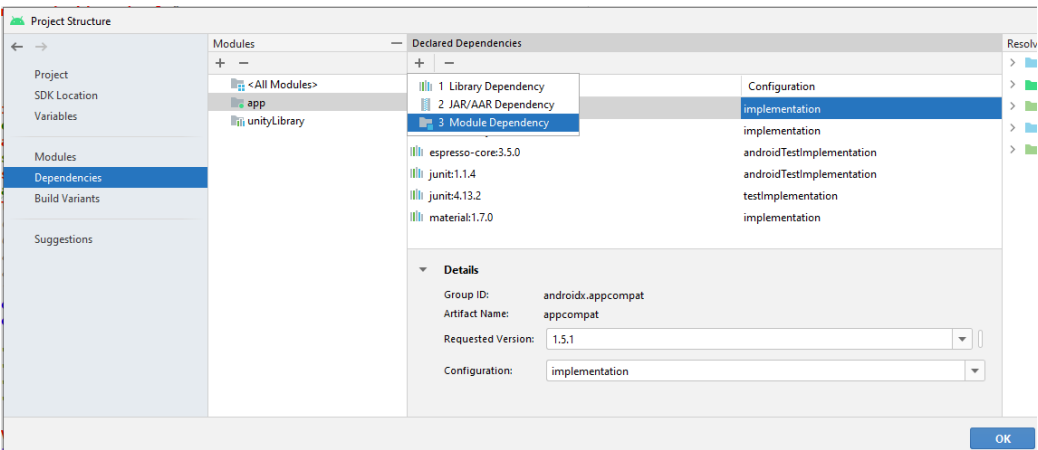


## 2.3 选择 unityLibrary 导入。点击 Finish



## 2.4 导入之后，为 Android 添加 unityLibrary 的引用

- 左上角 File——>Project Structure...
- 选择 Dependencies ——> app，然后点击右边这个加号 +，选择第三个 Module Dependency



- 勾选刚刚导入的 unity，点击 OK。再点击上图的 OK。

## 2.5 配置 Android 以及 unity 的 build.gradle 文件

- 将 SDK 配置成当前 Android 版本可以运行。Android 以及 unity 的 SDK 确保要一样，不然会报错，比如这个 minsdk。Build 无误就算是导入完成了！

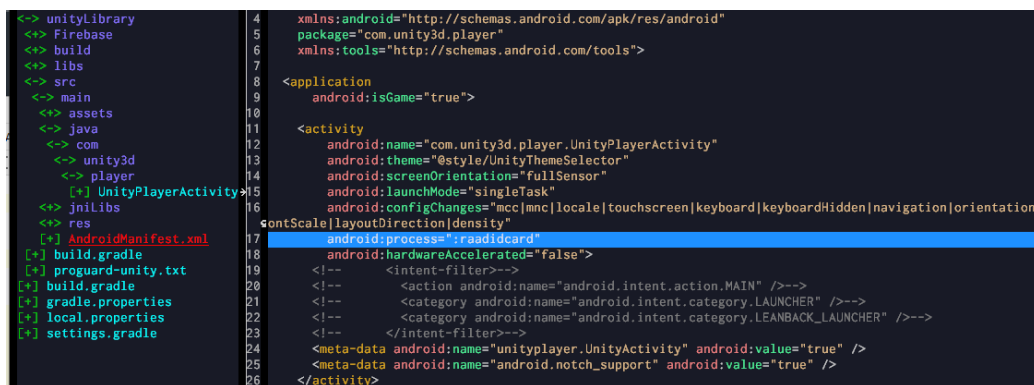
- 这里作些简单的版本修改适配自己的手机, 到项目可以构建成功为止.

## 3 Android 启动运行 unity

### 3.1 在 unity 的 AndroidMainfest.xml 文件

- 把 <intent-filter>-> 删掉或者注释掉, 留着的话, 当我们把程序运行到手机或者模拟机上时会有两个图标。
- 其次是在 <activity> 里加入这行代码, 实现多线程, 避免在从 unity 返回 Android 时也将 Android 界面也结束了。

android:process=":raadidcard"



### 3.2 在 app 的 AndroidMainfest.xml 文件里, 在图中位置加入这两行代码:

xmlns:tools="http://schemas.android.com/tools"

tools:replace="android:icon,android:theme,android:allowBackup"

- 可以成片复制的代码如下:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.unity3d.player">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        tools:replace="android:icon,android:theme,android:allowBackup"
        android:theme="@style/Theme.Test"
        tools:targetApi="31">

        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
            <meta-data
                android:name="android.app.lib_name"
                android:value="" />
            </activity>

        </application>
    </manifest>
```

<pre> &lt;-&gt; app &lt;-&gt; build &lt;-&gt; libs &lt;-&gt; src &lt;-&gt; androidTest &lt;-&gt; main   &lt;-&gt; java   &lt;-&gt; res   [+][AndroidManifest.xml] &lt;-&gt; test   [+][build.gradle] &lt;-&gt; gradle &lt;-&gt; unityLibrary &lt;-&gt; Firebase &lt;-&gt; build &lt;-&gt; libs &lt;-&gt; src   &lt;-&gt; main   &lt;-&gt; assets   &lt;-&gt; java   &lt;-&gt; com   &lt;-&gt; unity3d   &lt;-&gt; player   [+][UnityPlayerActivity] &lt;-&gt; jnilibs &lt;-&gt; res   [+][AndroidManifest.xml]   [+][build.gradle]   [+][proguard-unity.txt] </pre>	<pre> 2 &lt;manifest xmlns:android="http://schemas.android.com/apk/res/android" 3 4   xmlns:tools="http://schemas.android.com/tools" 5   package="com.unity3d.player"&gt; 6 7       &lt;application 8   android:allowBackup="true" 9       android:dataExtractionRules="@xml/data_extraction_rules" 10      android:fullBackupContent="@xml/backup_rules" 11      android:icon="@mipmap/ic_launcher" 12      android:label="@string/app_name" 13      android:roundIcon="@mipmap/ic_launcher_round" 14      android:supportRtl="true" 15      tools:replace="android:icon,android:theme,android:allowBackup" 16      android:theme="@style/Theme.Test" 17      tools:targetApi="31"&gt; 18          &lt;activity 19              android:name=".MainActivity" 20              android:exported="true"&gt; 21              &lt;intent-filter&gt; 22                  &lt;action android:name="android.intent.action.MAIN" /&gt; 23 24                  &lt;category android:name="android.intent.category.LAUNCHER" /&gt; 25              &lt;/intent-filter&gt; 26 27              &lt;meta-data 28                  android:name="android.app.lib_name" 29                  android:value="" /&gt; 30          &lt;/activity&gt; </pre>
---	--

### 3.3 在 app 的 build.gradle 里加入这行代码。

```

ndk {
    abiFilters 'armeabi-v7a'
}

```

<pre> &lt;-&gt; app &lt;-&gt; build &lt;-&gt; libs &lt;-&gt; src &lt;-&gt; androidTest &lt;-&gt; main   &lt;-&gt; java   &lt;-&gt; res   [+][AndroidManifest.xml] &lt;-&gt; test   [+][build.gradle] &lt;-&gt; gradle &lt;-&gt; unityLibrary &lt;-&gt; Firebase &lt;-&gt; build &lt;-&gt; libs &lt;-&gt; src   &lt;-&gt; main   &lt;-&gt; assets   &lt;-&gt; java   &lt;-&gt; com   &lt;-&gt; unity3d   &lt;-&gt; player   [+][UnityPlayerActivity] &lt;-&gt; jnilibs &lt;-&gt; res   [+][AndroidManifest.xml] </pre>	<pre> 2 id 'com.android.application' 3 } 4 5 android { 6     namespace 'com.unity3d.player' 7     compileSdk 32 8 9     defaultConfig { 10         applicationId "com.unity3d.player" 11         minSdk 25 12         targetSdk 31 13         versionCode 1 14         versionName "1.0" 15 16         ndk { 17             abiFilters 'armeabi-v7a' 18         } 19 20         testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner" 21     } 22 23     buildTypes { 24         release { 25             minifyEnabled false 26             proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro' 27         } 28     } 29 } </pre>
---	--

### 3.4 在 app 的 main->res->values->strings.xml 里加入这行代码

- 都还没有去想, 这句话能起到什么作用, 应该是关系不大, 或是可以跳过绕过的小细节

```
<string name="game_view_content_description">Game view</string>
```

```

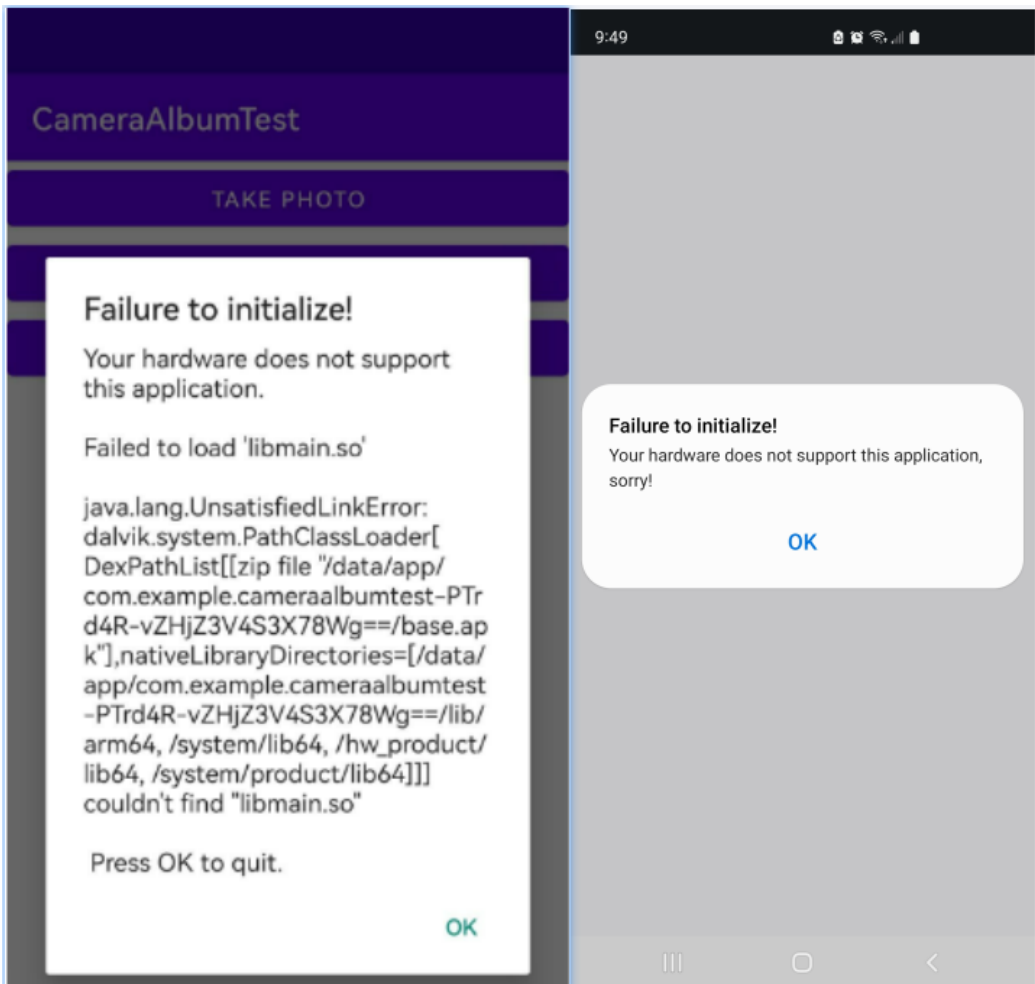
<-> app
<-> build
<-> libs
<-> src
<-> androidTest
<-> main
<-> java
<-> res
    <-> drawable
    <-> drawable-v24
    <-> layout
    <-> mipmap-anydpi-v26
    <-> mipmap-hdpi
    <-> mipmap-mdpi
    <-> mipmap-xhdpi
    <-> mipmap-xxhdpi
    <-> mipmap-xxxhdpi
    <-> values
    [+ colors.xml
    [+ strings.xml
    [+ themes.xml
2
3
4
5
6
7
<string name="app_name">test</string>

<string name="game_view_content_description">Game view</string>

</resources>

```

- 进行这两步操作的原因是，我在运行到手机时，他显示硬件不支持或者闪退。加入上面两个代码后就可以正常启动 unity。
- 我个人认为真正起作用的是上一步关于手机架构的设置 ndk 那三行，与上面字符串无关，应该是无关的



### 3.5 点击按钮启动 unity(画蛇添足)

- 感觉这个连接过程对于自己的项目就是画蛇添足. 可是如何既能避开这一步, 又能两者很好的平滑交互呢? 对于现在的自己, 是个问题和挑战
- 在主工程的 activity\_main.xml 文件里添加一个按钮. MainActivity.java 里加入启动事件, 如果在这里 layout 标红的话, 就把鼠标移到 layout 下面, 建立一个 layout 就行, 我分析是主工程的问题, 这个影响不大

<Button

```
android:id="@+id/showUnityBtn"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="Show Unity"/>
```

```
f:/tmp/test/
<?xml version="1.0" encoding="utf-8"?>
<!-- <androidx.constraintlayout.widget.ConstraintLayout -->
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".MainActivity">
  <!-- <TextView -->
  <!-- android:layout_width="wrap_content" -->
  <!-- android:layout_height="wrap_content" -->
  <!-- android:text="Hello World!" -->
  <!-- app:layout_constraintBottom_toBottomOf="parent" -->
  <!-- app:layout_constraintEnd_toEndOf="parent" -->
  <!-- app:layout_constraintStart_toStartOf="parent" -->
  <!-- app:layout_constraintTop_toTopOf="parent" /> -->
  <Button
    android:id="@+id/showUnityBtn"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Show Unity"/>
</LinearLayout>
<!-- </androidx.constraintlayout.widget.ConstraintLayout -->
```

- MainActivity.cs 里的回调设置

```
Button btn = (Button)findViewById(R.id.showUnityBtn);
btn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
```

```
// <----- UnityPlayerActivity <= com.unity3d.player 这里就是刚刚那个包名奇怪的地方, 要不然 找不到 下面的 UnityPlayerActivity
Intent intent = new Intent(MainActivity.this, UnityPlayerActivity.class); // <----- UnityPlayerActivity
startActivity(intent);
}
});
```

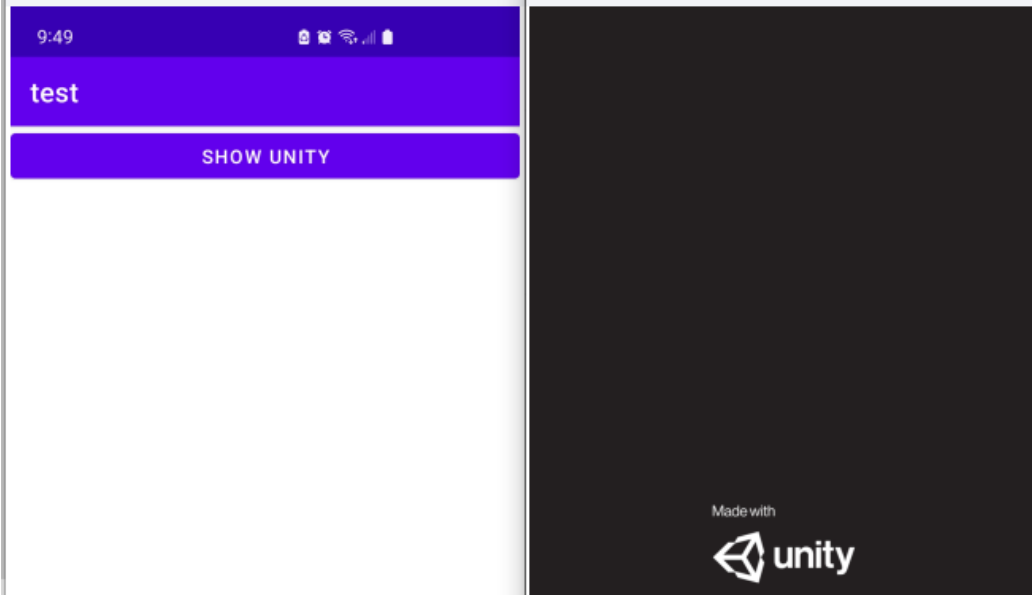
```
f:/tmp/test/
package com.unity3d.player; // <-----
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity {
    @Override protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Button btn = (Button)findViewById(R.id.showUnityBtn);
        btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                // <----- UnityPlayerActivity <= com.unity3d.player 这里就是刚刚那个包名奇怪的地方, 要不然 找不到 下面的 UnityPlayerActivity 类
                Intent intent = new Intent(MainActivity.this, UnityPlayerActivity.class); // <----- UnityPlayerActivity
                startActivity(intent);
            }
        });
    }
}
```



## 4 启动运行



## 5 那么现在就是说: 安卓 SDK 与 unity 的交互与打包基本没有问题了

- 但对自己更大的挑战是: 为什么 unity 里一个空物件挂载到热更新的过程, 我打包之后在安卓手机上运行不出来, 仍需要时间 debug 这个过程
- 过程中遇到过, 还会遇到很多不懂的问题, 比如同样的某些 android studio 里加 android:exported="true" 各种标签等, 如果只用 unity 打包, 该如何实现呢? 两套不同的打包机制都得弄明白. 但都是这么一个学习的过程, 不会被轻易挫败.
- 相比之下, 安卓 SDK 的实现极其简单, 可以放在后面, 等这些疑难杂症都解决放心了, 再去写简单一点儿的