# 接入说明 - 新

## Android接入说明

### 权限要求

**要求应用app必须具有以下权限**

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission
android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
```

### API接口描述

```
package com.netshield;

public class Netshield {
    /**
     * 获取客户端IP
     * @return 客户端IP
     */
    public static String GetClientIP();
}
```

### 接入说明

#### 代码接入

AndroidManifest.xml appliction节点中添加以下代码：
其中 **${应用包名}** 需要参考实际应用包名（manifest节点下的package属性）

```
<provider android:authorities="${应用包名}" android:exported="false"
android:name="com.netshield.NetshieldContentProvider"/>
```

例如：

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.test.app">
    <application ...>
        ...
        <provider android:authorities="com.test.app"
android:exported="false"
```

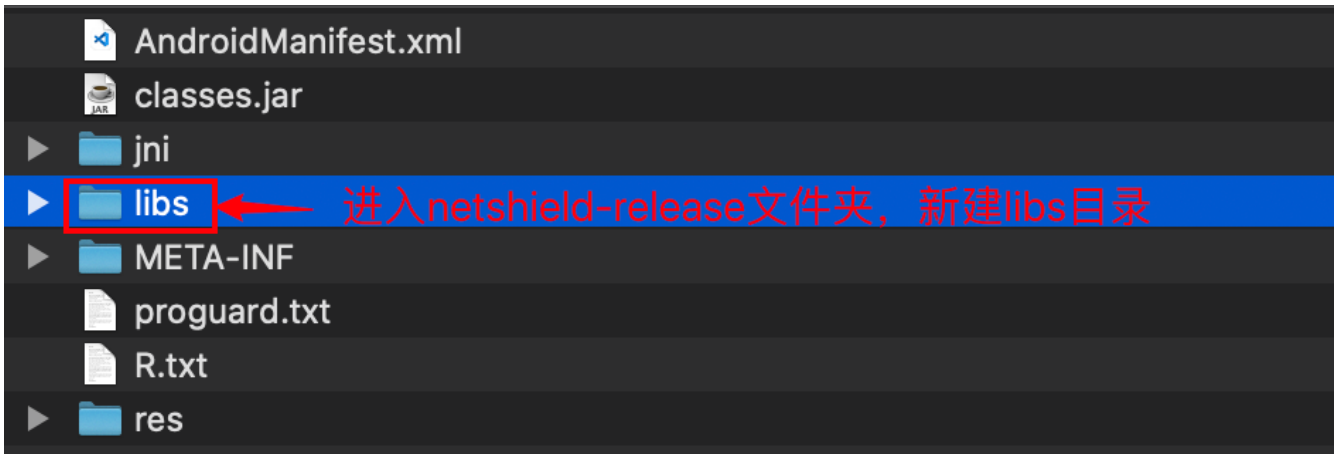
```
android:name="com.netshield.NetshieldContentProvider"/>
    </application>
</manifest>
```

## 原生项目

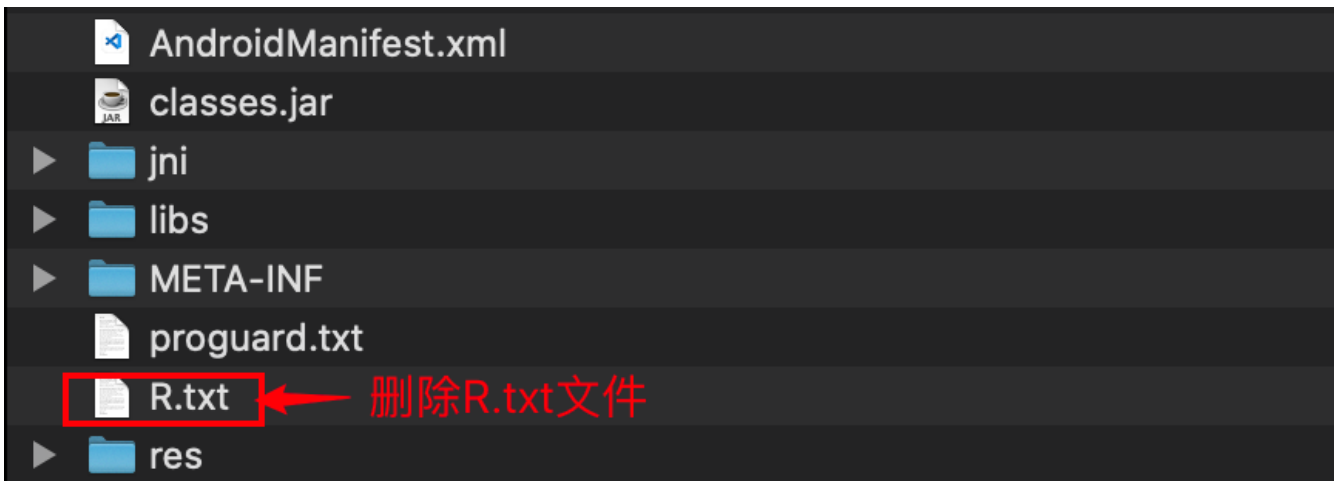
### Eclipse

1. 将 AAR 文件后缀名改为 ZIP 并解压，解压的目录需要和项目在同一盘符下。比如项目在F盘，则解压的sdk也应在 F盘。

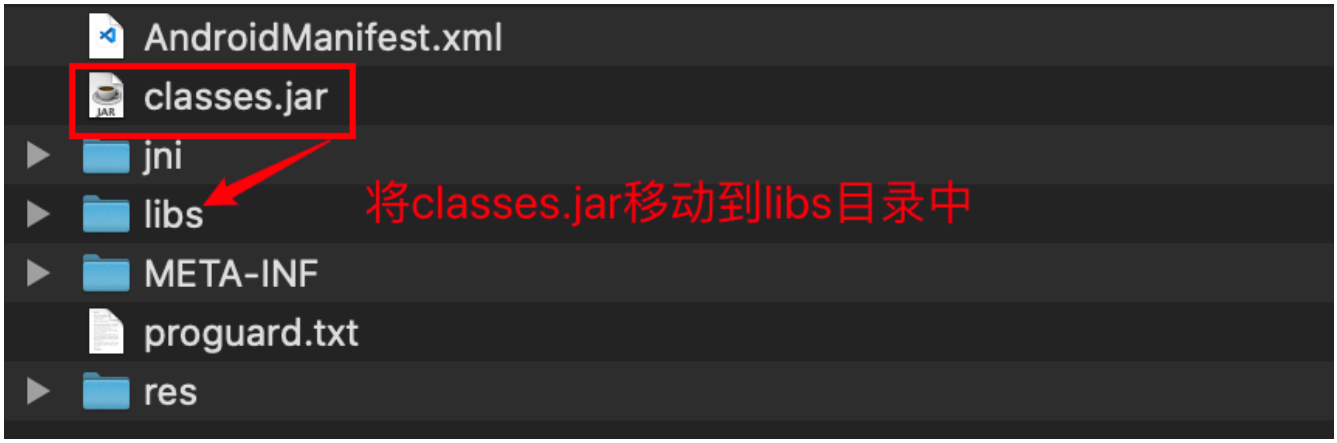

2. 在解压得到的目录里创建一个 libs 文件夹



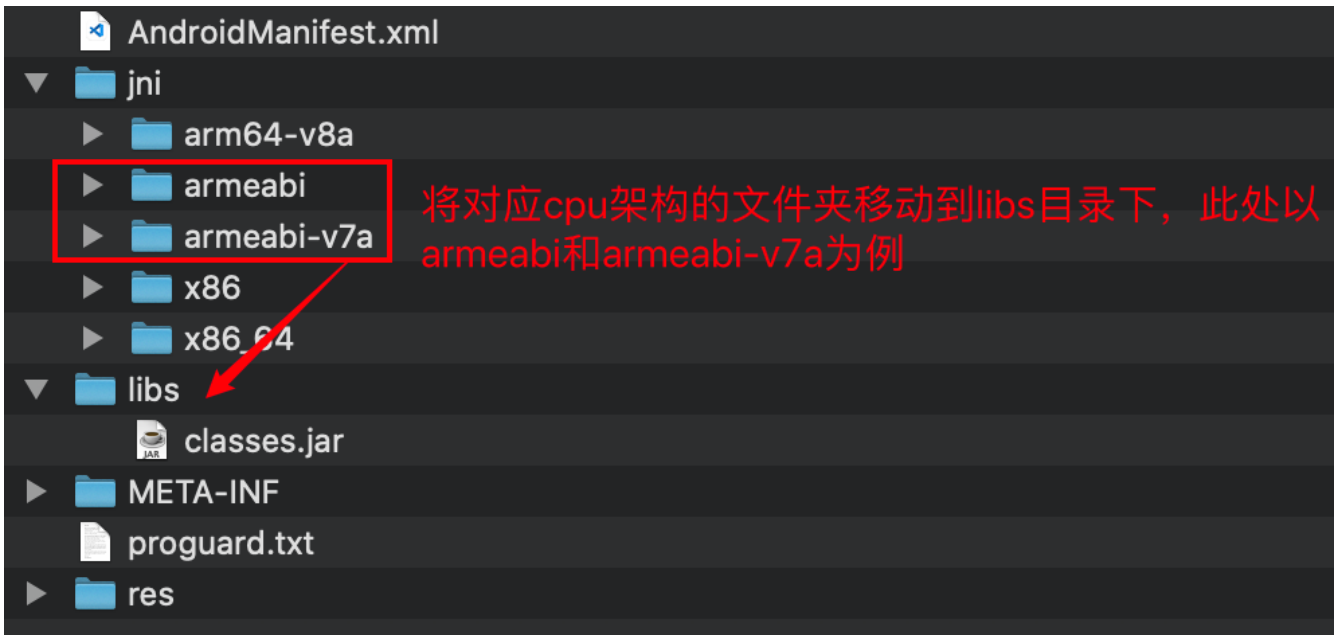

3. 删除 R.txt文件



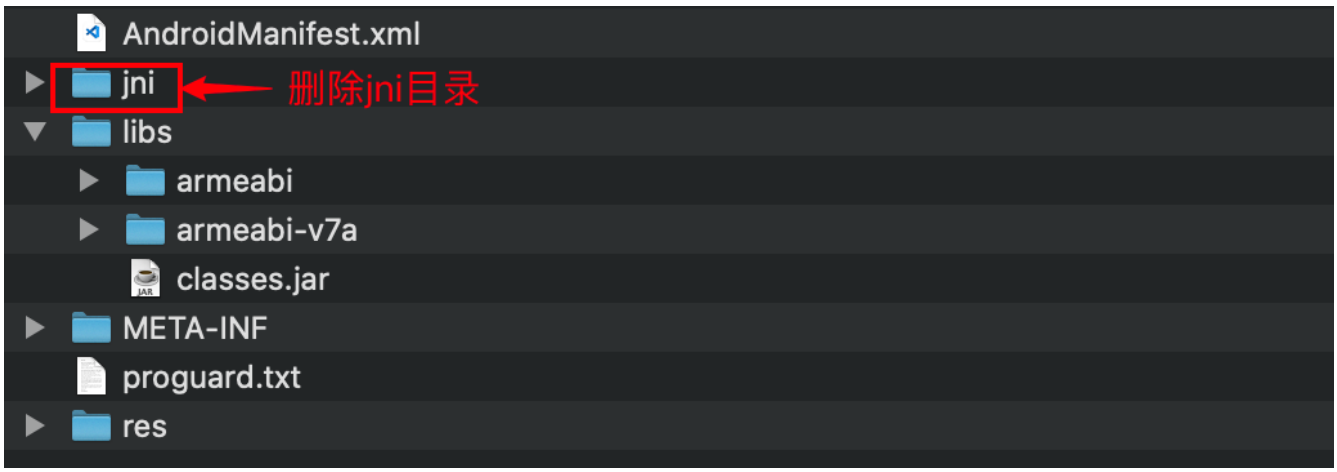
4. 将classes.jar移动到libs文件夹下




5. 将jni下和项目架构相同的文件夹(**比如项目用到了armeabi和armeabi-v7a，则只拷贝armeabi和armeabi- v7a，其他的不要动**)移到 libs 文件夹下

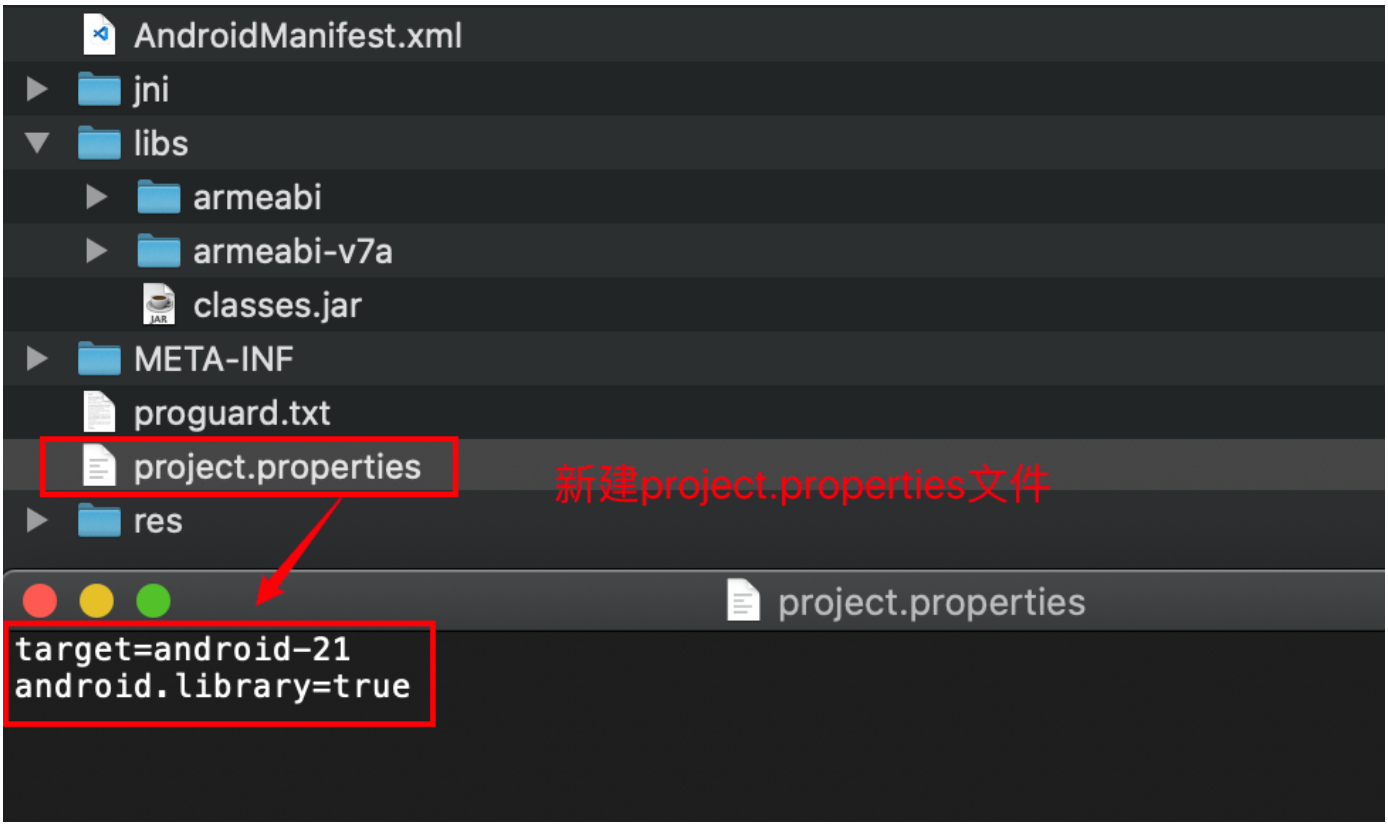



6. 删除jni文件夹

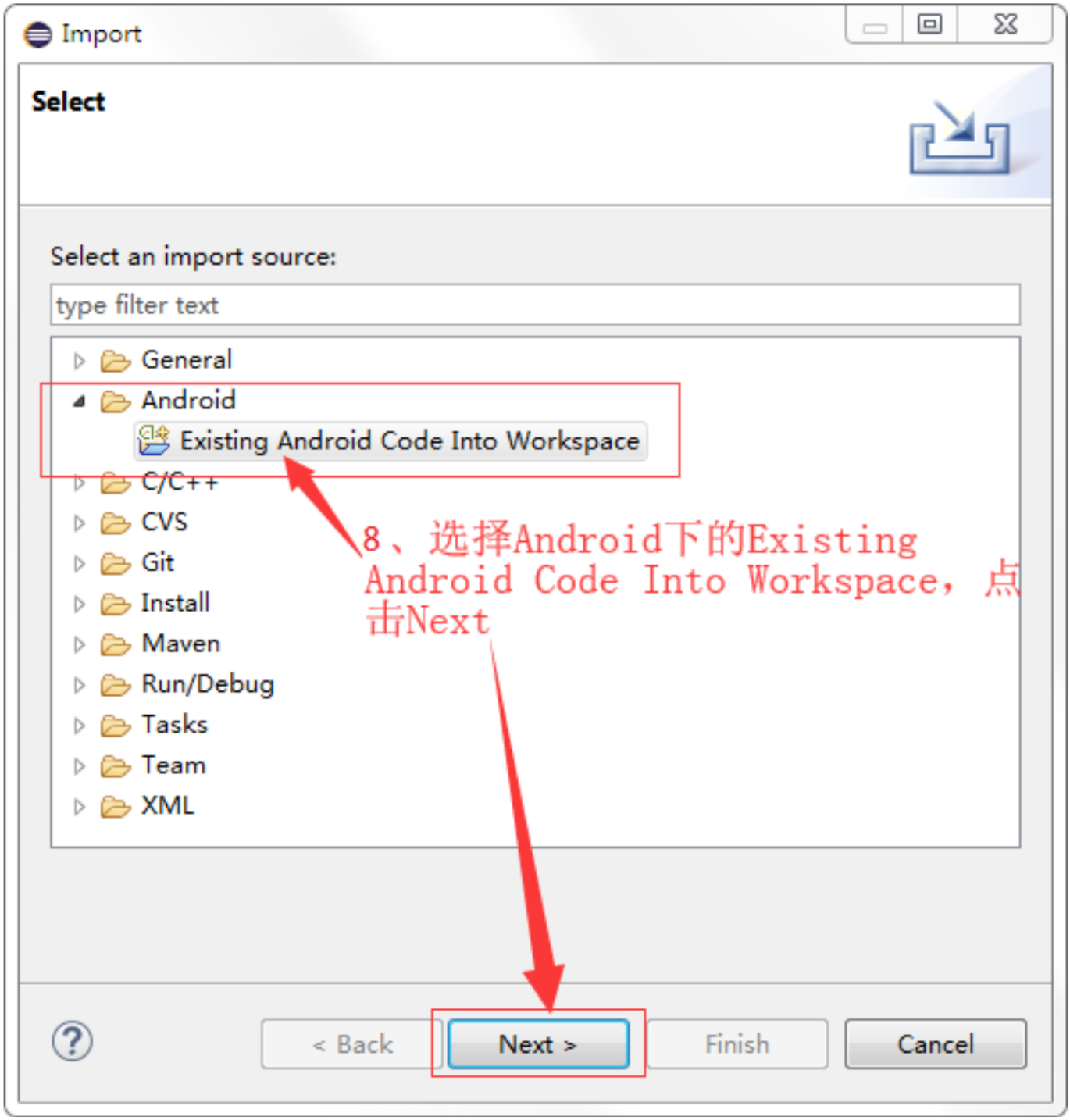


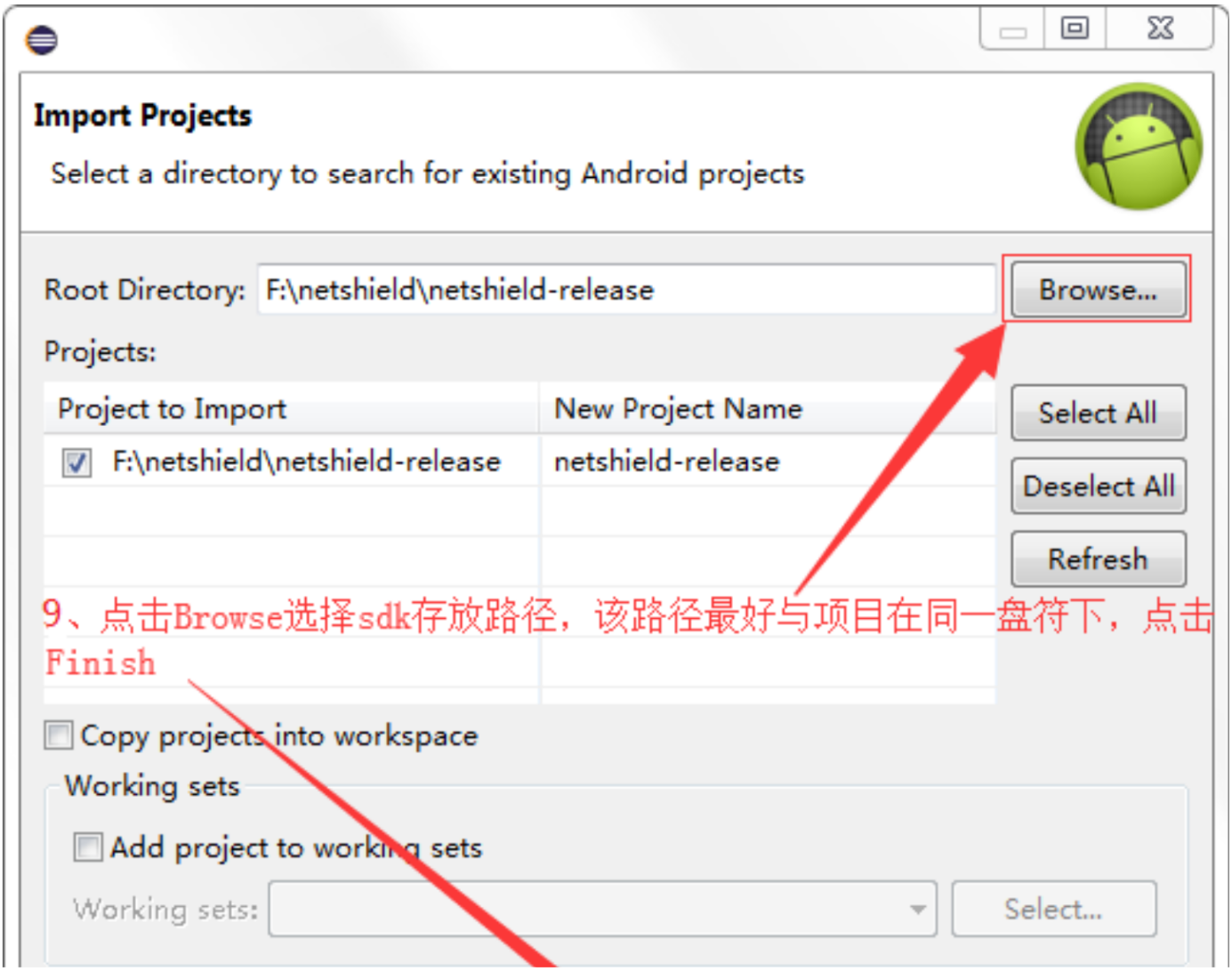

7. 在解压得到的目录里创建 project.properties 文件，内容如下：

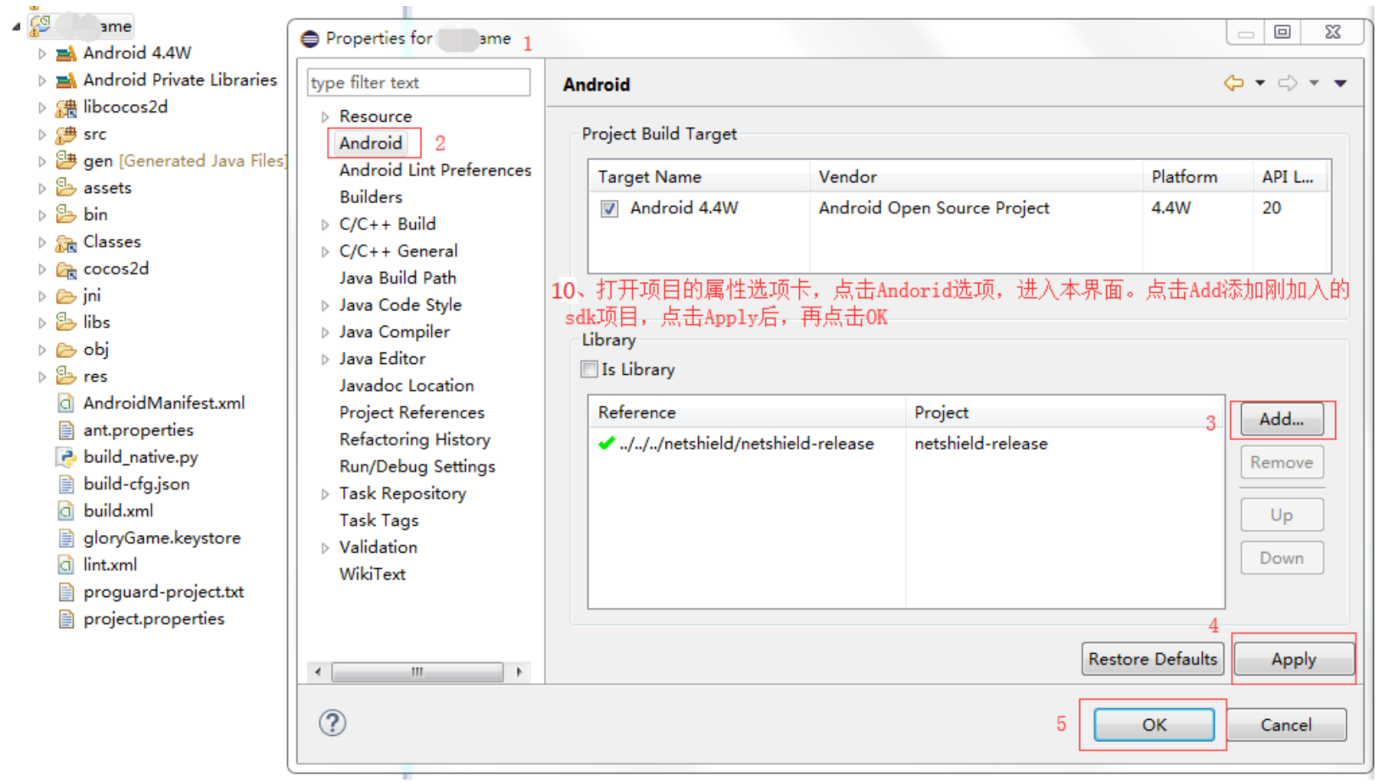
```
target=android-21
android.library=true
```




8. 导入sdk到eclipse中


Import
Select
Select an import source:
type filter text
General
Android
Existing Android Code Into Workspace
C/C++
CVS
Git
Install
Maven
Run/Debug
Tasks
Team
XML
8、选择Android下的Existing Android Code Into Workspace，点击Next
< Back
Next >
Finish
Cancel




## Android Studio

如果项目开启了prguard优化，请添加以下规则

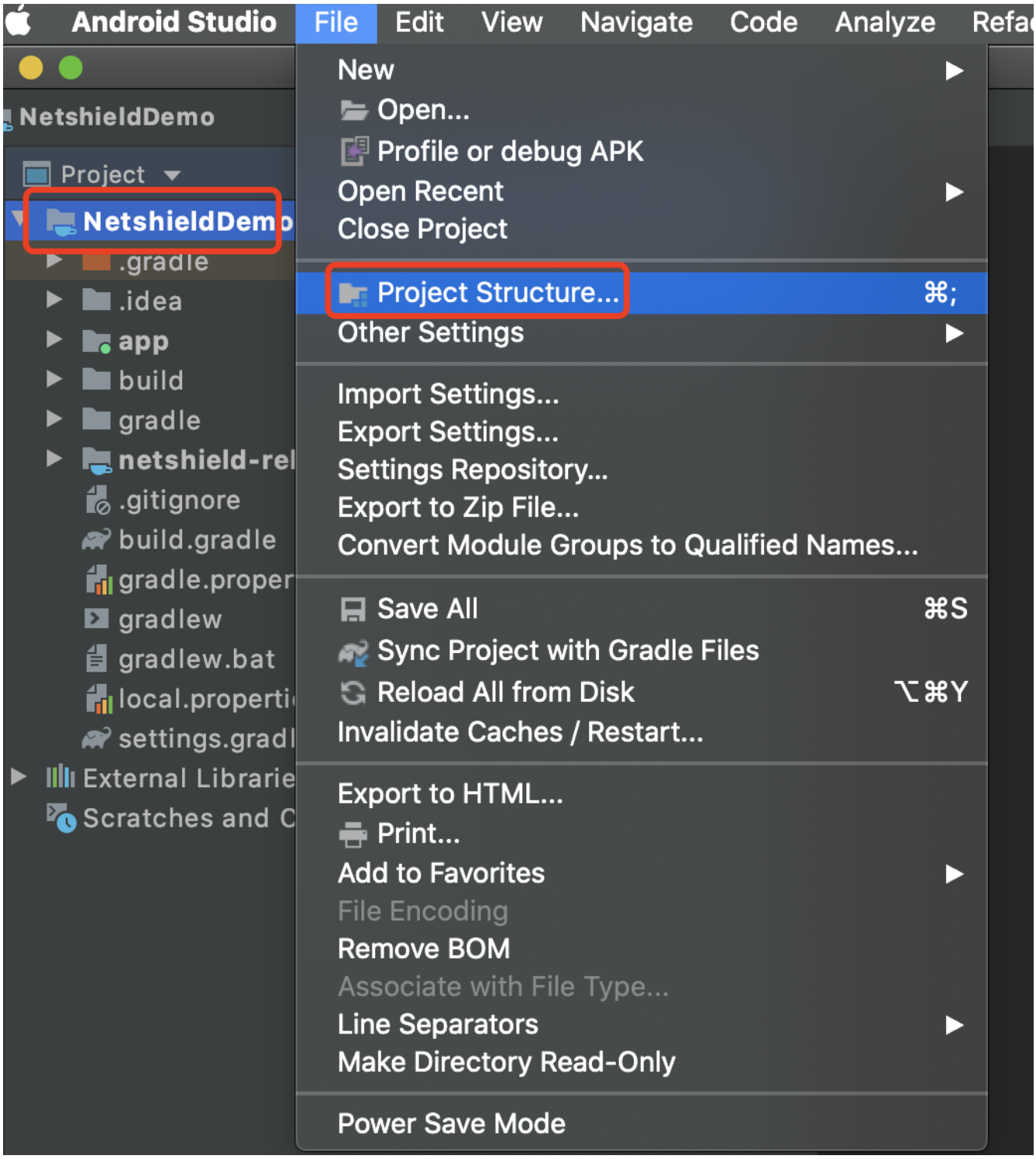
```
-keep class com.netshield.* { *; }
-dontwarn com.tencent.bugly.**
-keep public class com.tencent.bugly.**{*;}
```

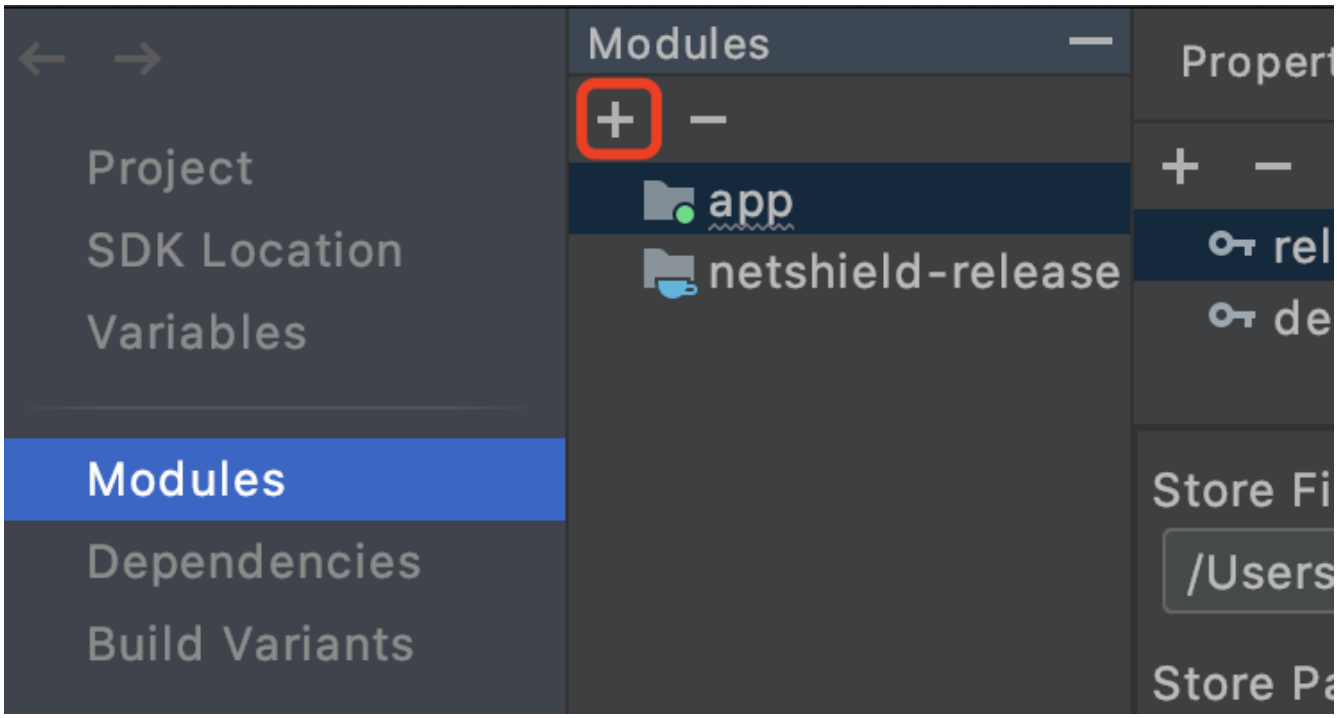
**如果报错transformClassesAndResourcesWithProguardForRelease，请添加**

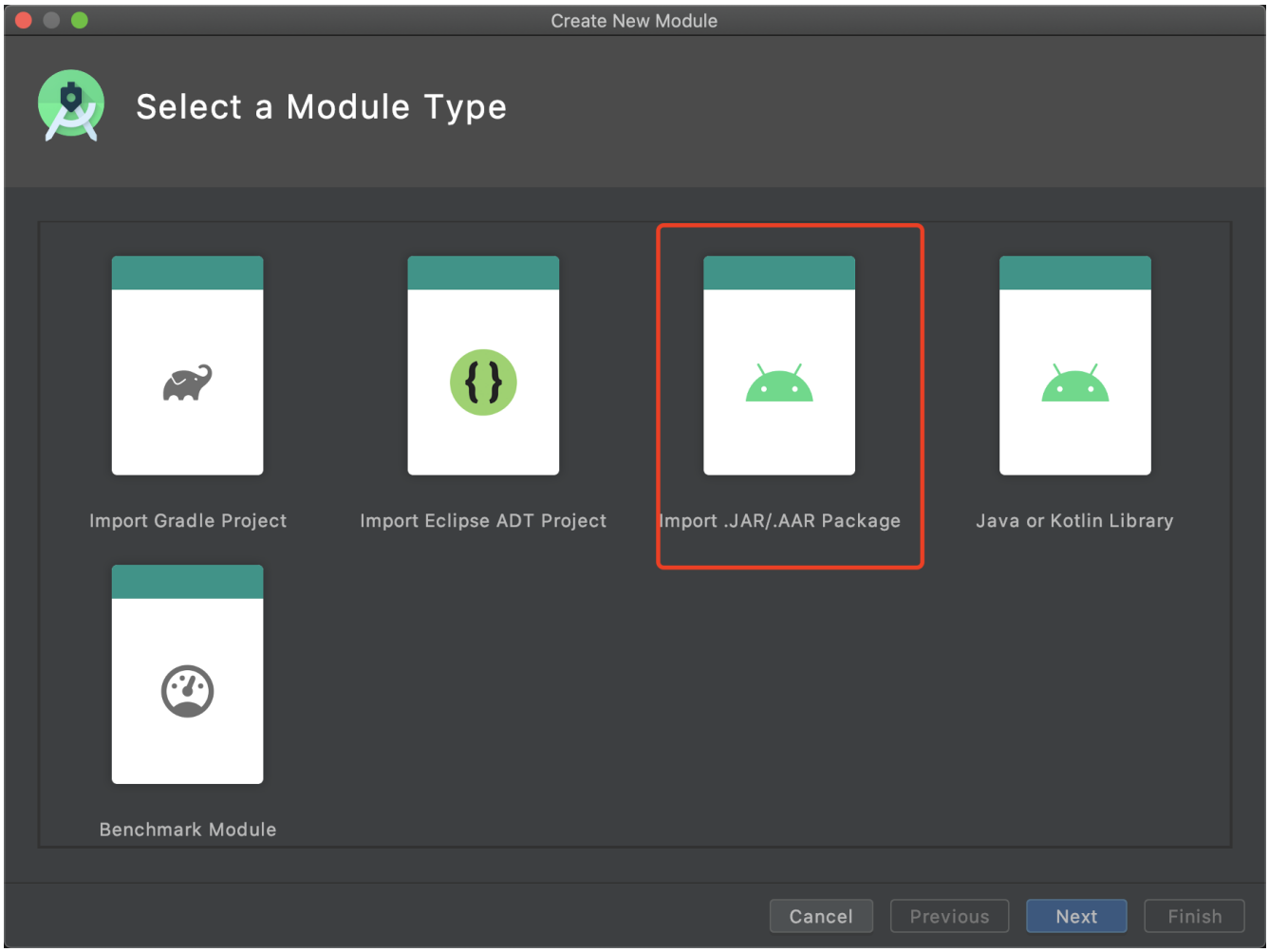
```
-ignorewarnings
```

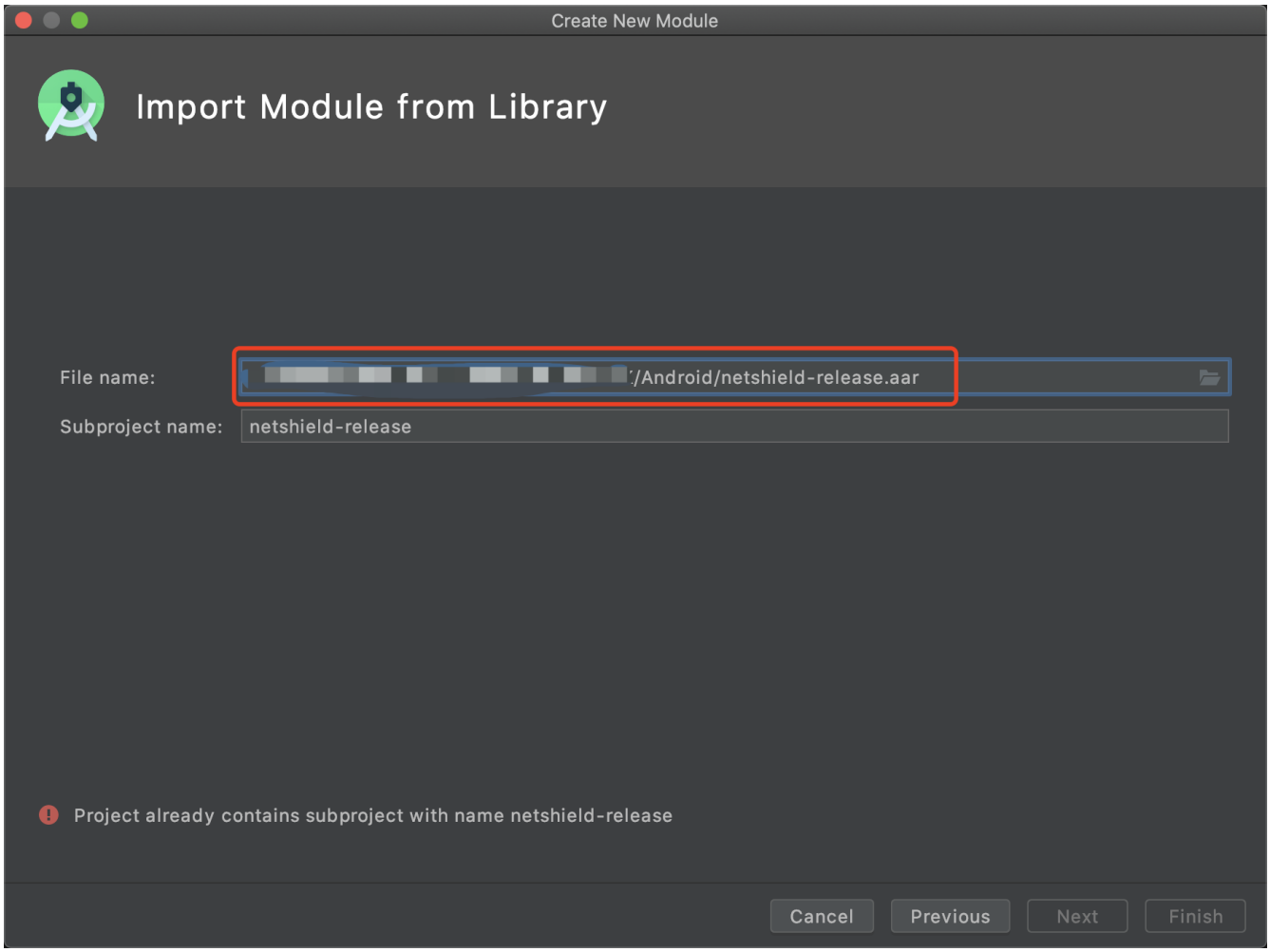
假设原module名为app，新增的module名为netshield

1. 添加netshield-release模块

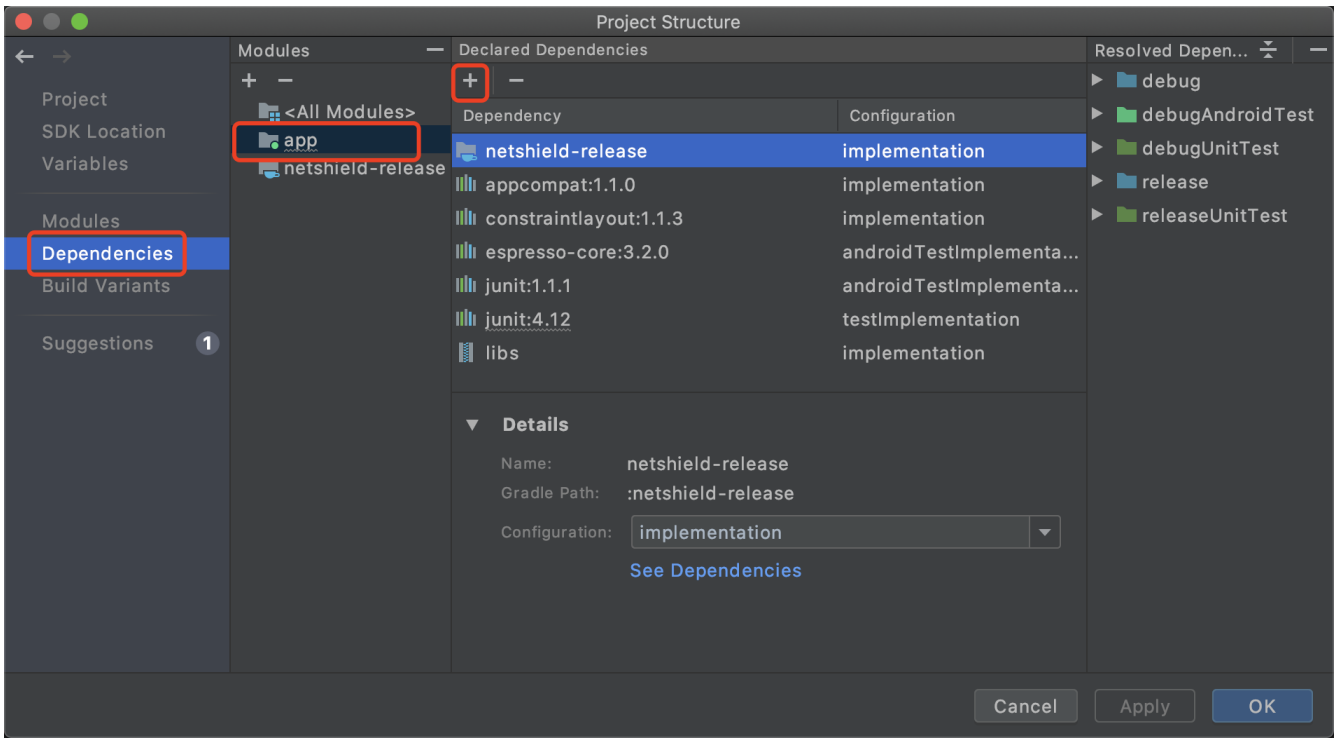


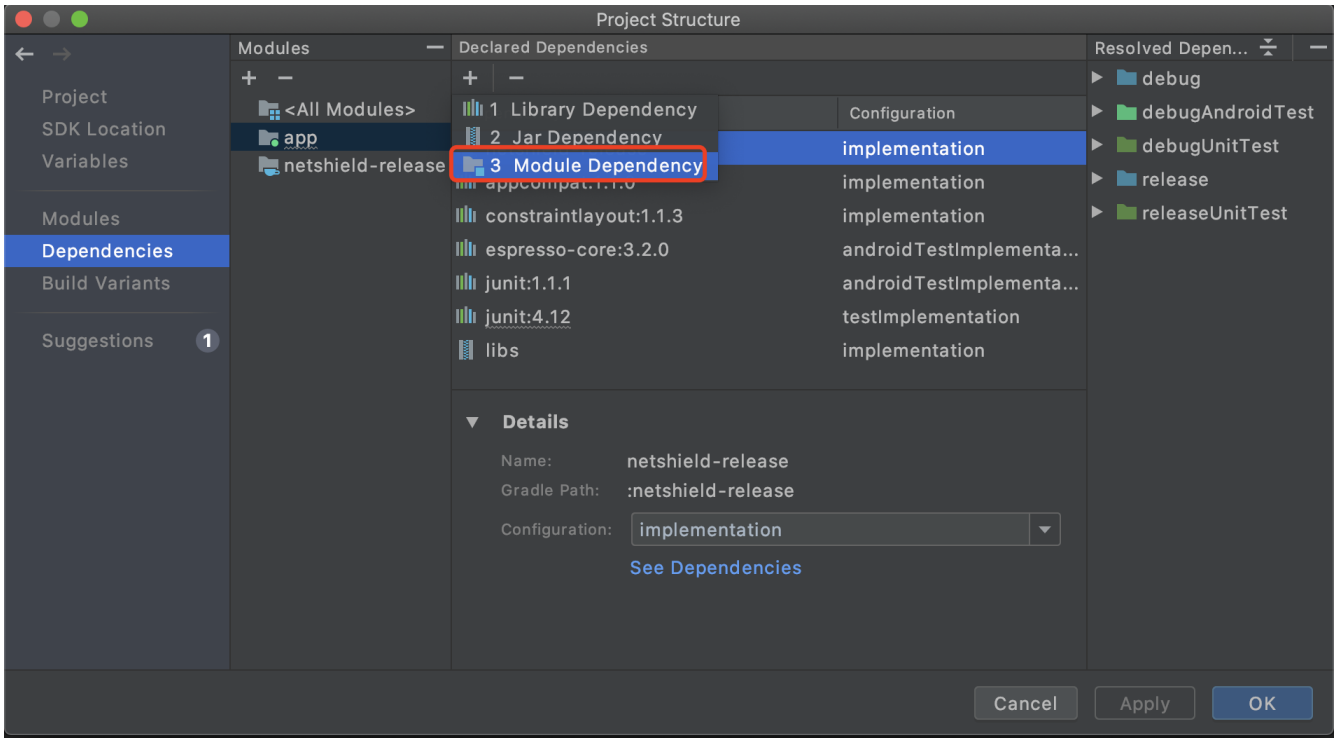

Project
SDK Location
Variables
Modules
Dependencies
Build Variants
Modules
app
netshield-release

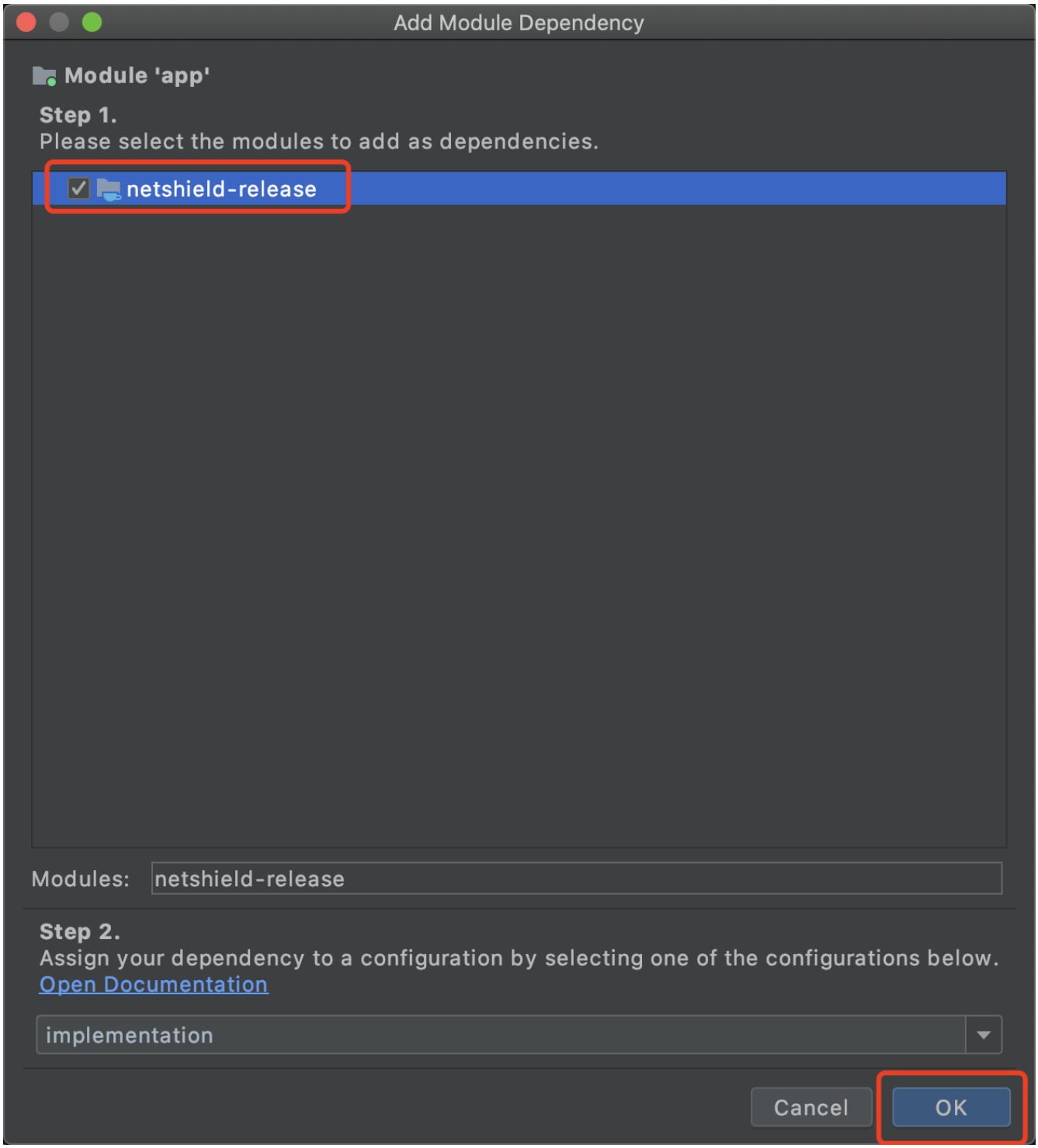

Create New Module
Select a Module Type
Import Gradle Project
Import Eclipse ADT Project
Import .JAR/.AAR Package
Java or Kotlin Library
Benchmark Module
Cancel
Previous
Next
Finish


Create New Module
Import Module from Library
File name:
/Android/netshield-release.aar
Subproject name:
netshield-release
Project already contains subproject with name netshield-release
Cancel
Previous
Next
Finish

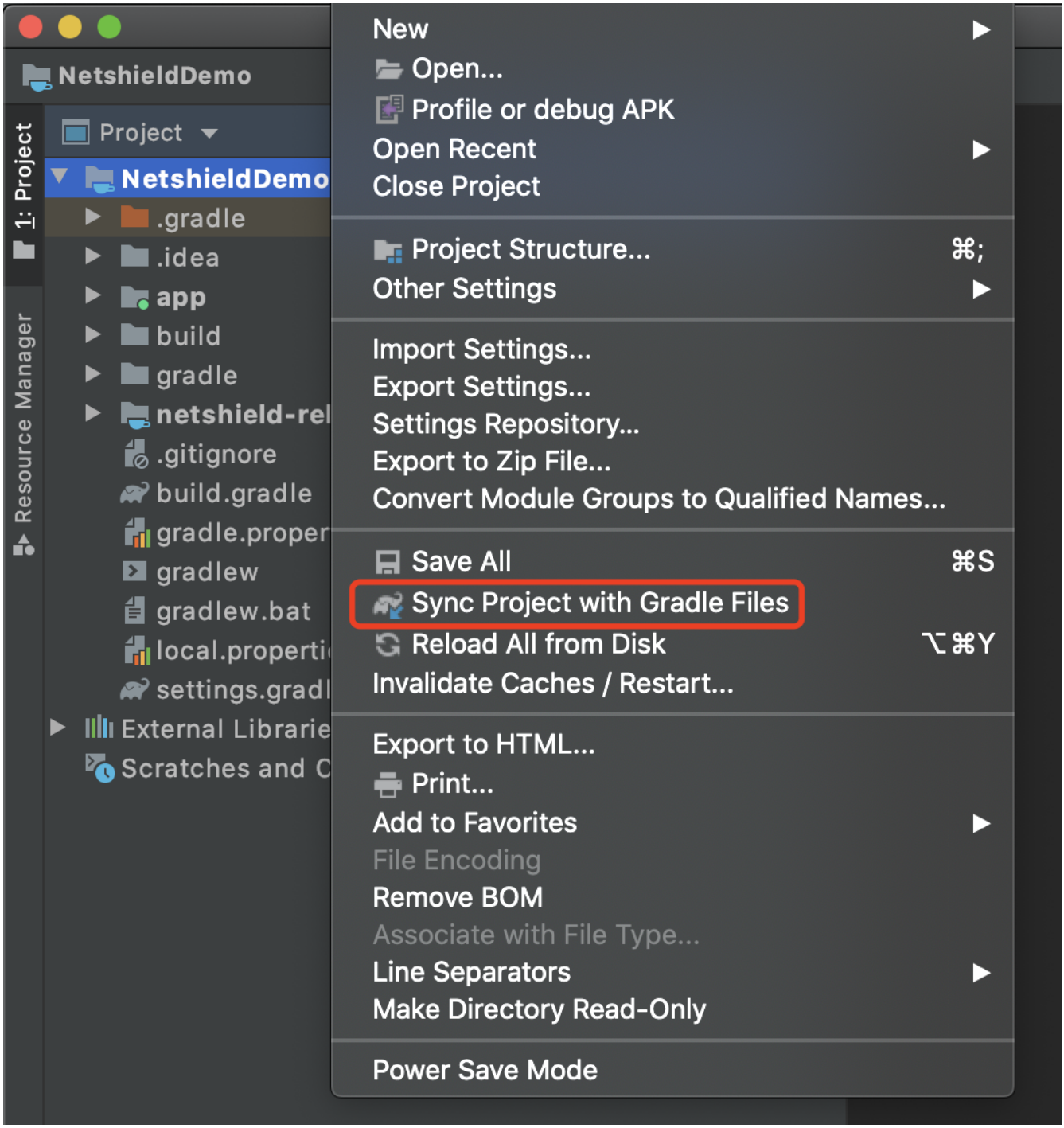
2. 设置项目依赖


Project Structure
Project
SDK Location
Variables
Modules
Dependencies
Build Variants
Suggestions
1
Modules
<All Modules>
app
netshield-release
Declared Dependencies
Dependency
Configuration
netshield-release
implementation
appcompat:1.1.0
implementation
constraintlayout:1.1.3
implementation
espresso-core:3.2.0
androidTestImplementa...
junit:1.1.1
androidTestImplementa...
junit:4.12
testImplementation
libs
implementation
Details
Name:
netshield-release
Gradle Path:
:netshield-release
Configuration:
implementation
See Dependencies
Resolved Depen...
debug
debugAndroidTest
debugUnitTest
release
releaseUnitTest
Cancel
Apply
OK


Project Structure
Project
SDK Location
Variables
Modules
Dependencies
Build Variants
Suggestions
1
Modules
<All Modules>
app
netshield-release
Declared Dependencies
1 Library Dependency
2 Jar Dependency
3 Module Dependency
Configuration
implementation
implementation
constraintlayout:1.1.3
implementation
espresso-core:3.2.0
androidTestImplementa...
junit:1.1.1
androidTestImplementa...
junit:4.12
testImplementation
libs
implementation
Details
Name:
netshield-release
Gradle Path:
:netshield-release
Configuration:
implementation
See Dependencies
Resolved Depen...
debug
debugAndroidTest
debugUnitTest
release
releaseUnitTest
Cancel
Apply
OK


Add Module Dependency
Module 'app'
Step 1.
Please select the modules to add as dependencies.
netshield-release
Modules:
netshield-release
Step 2.
Assign your dependency to a configuration by selecting one of the configurations below.
Open Documentation
implementation
Cancel
OK

3. 同步配置

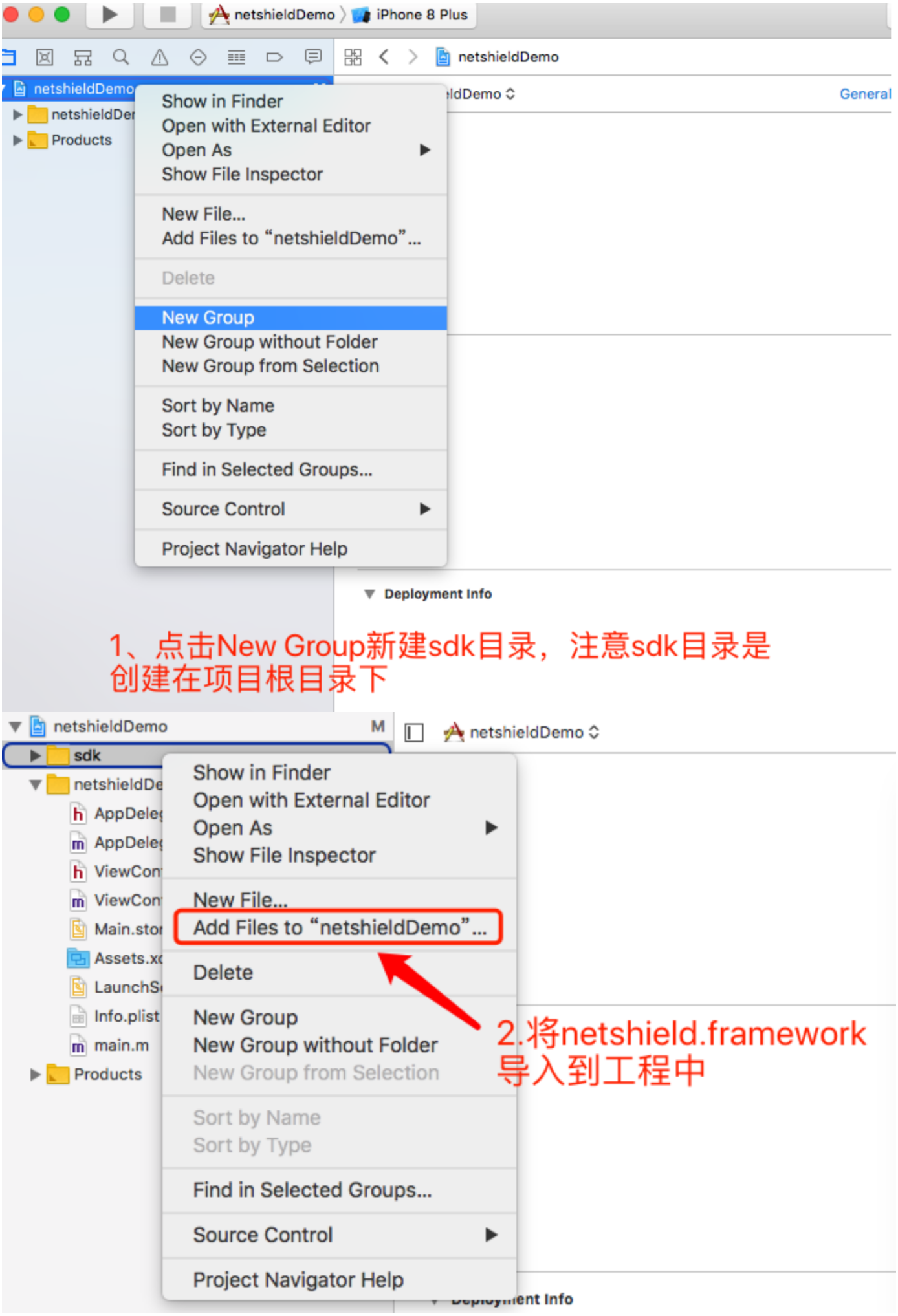


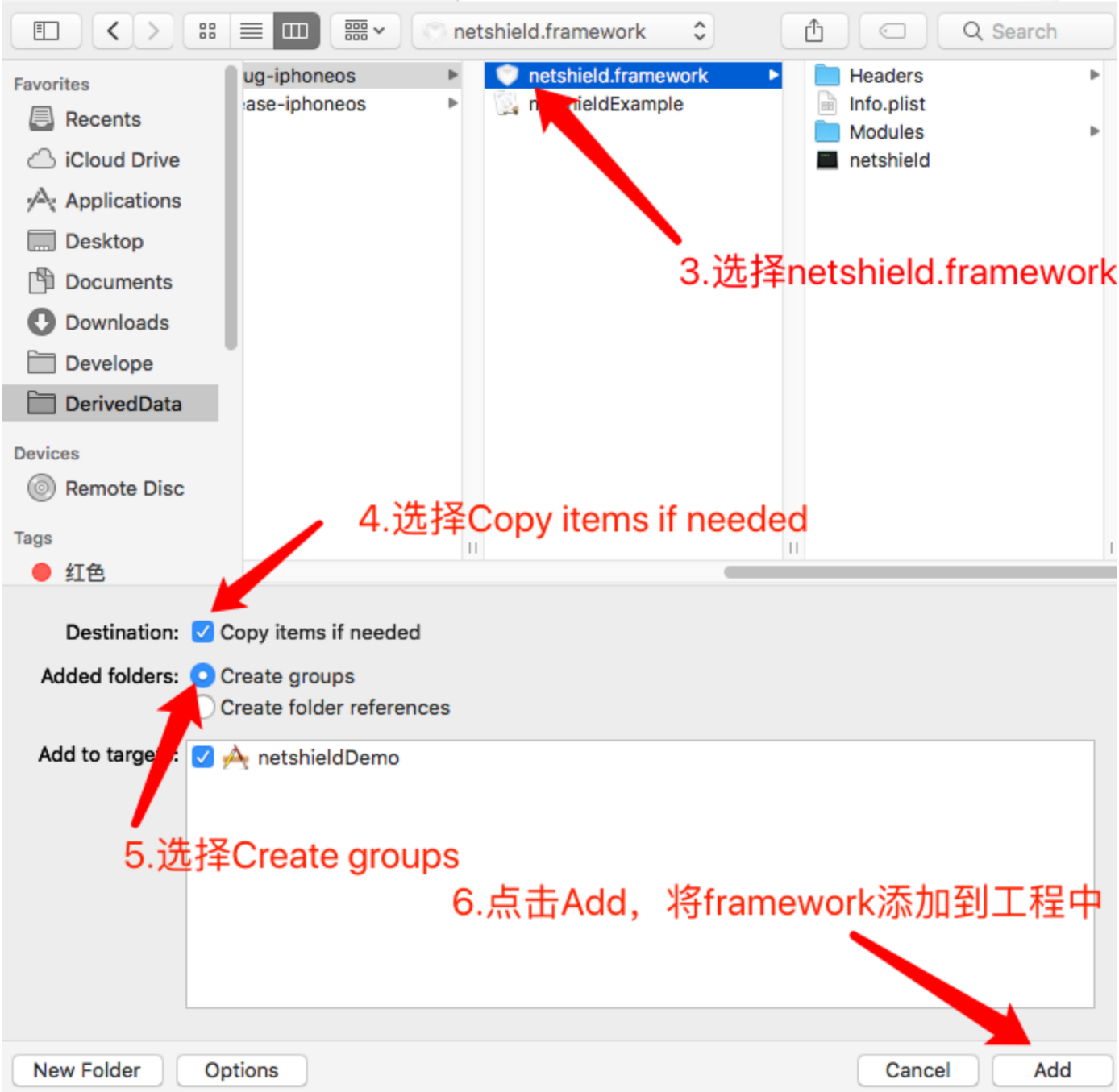

# IOS

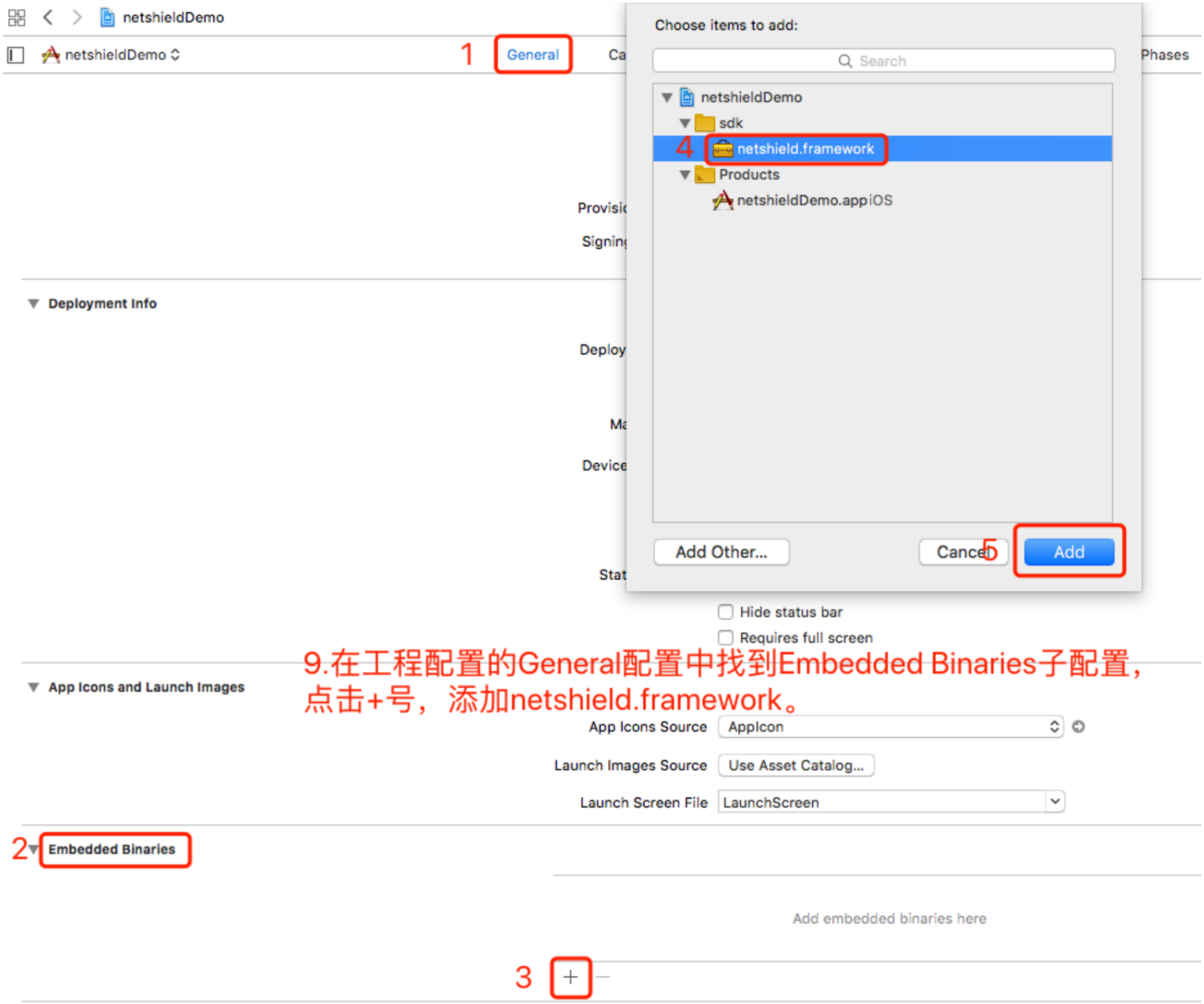
## API接口描述

```
/*!
 * 获取客户端IP，务必在 StartNetshieldService 函数后调用
 * @return NULL-获取失败，否则即为客户端IP
 */
const char* GetRealClientIP();
```

## 接入说明


netshieldDemo
iPhone 8 Plus
netshieldDemo
netshieldDemo
Products
Show in Finder
Open with External Editor
Open As
Show File Inspector
New File...
Add Files to “netshieldDemo”...
Delete
New Group
New Group without Folder
New Group from Selection
Sort by Name
Sort by Type
Find in Selected Groups...
Source Control
Project Navigator Help
General
Deployment Info
1、点击New Group新建sdk目录，注意sdk目录是创建在项目根目录下
netshieldDemo
sdk
Products
Show in Finder
Open with External Editor
Open As
Show File Inspector
New File...
Add Files to “netshieldDemo”...
Delete
New Group
New Group without Folder
New Group from Selection
Sort by Name
Sort by Type
Find in Selected Groups...
Source Control
Project Navigator Help
2.将netshield.framework导入到工程中


netshield.framework
Search
Favorites
Recents
iCloud Drive
Applications
Desktop
Documents
Downloads
Develope
DerivedData
Devices
Remote Disc
Tags
红色
ug-iphoneos
ase-iphoneos
netshield.framework
ieldExample
Headers
Info.plist
Modules
netshield
3.选择netshield.framework
4.选择Copy items if needed
Destination: Copy items if needed
Added folders: Create groups
Create folder references
Add to targe
netshieldDemo
5.选择Create groups
6.点击Add，将framework添加到工程中
New Folder
Options
Cancel
Add




注意Embed选项要选择**Embed & Sign**

# Windows

## API接口描述

```
//功能开启
_declspec(dllexport) BOOL StartNetshieldService();
//获取客户端IP，需要在 StartNetshieldService 后调用
_declspec(dllexport) const char* GetRealClientIP();
```

## Visual Studio项目

1. 拷贝netshield.dll到应用目录下
2. 修改main函数

```
int main()
{
    HINSTANCE pNetshield = NULL;
#ifdef UNICODE
```

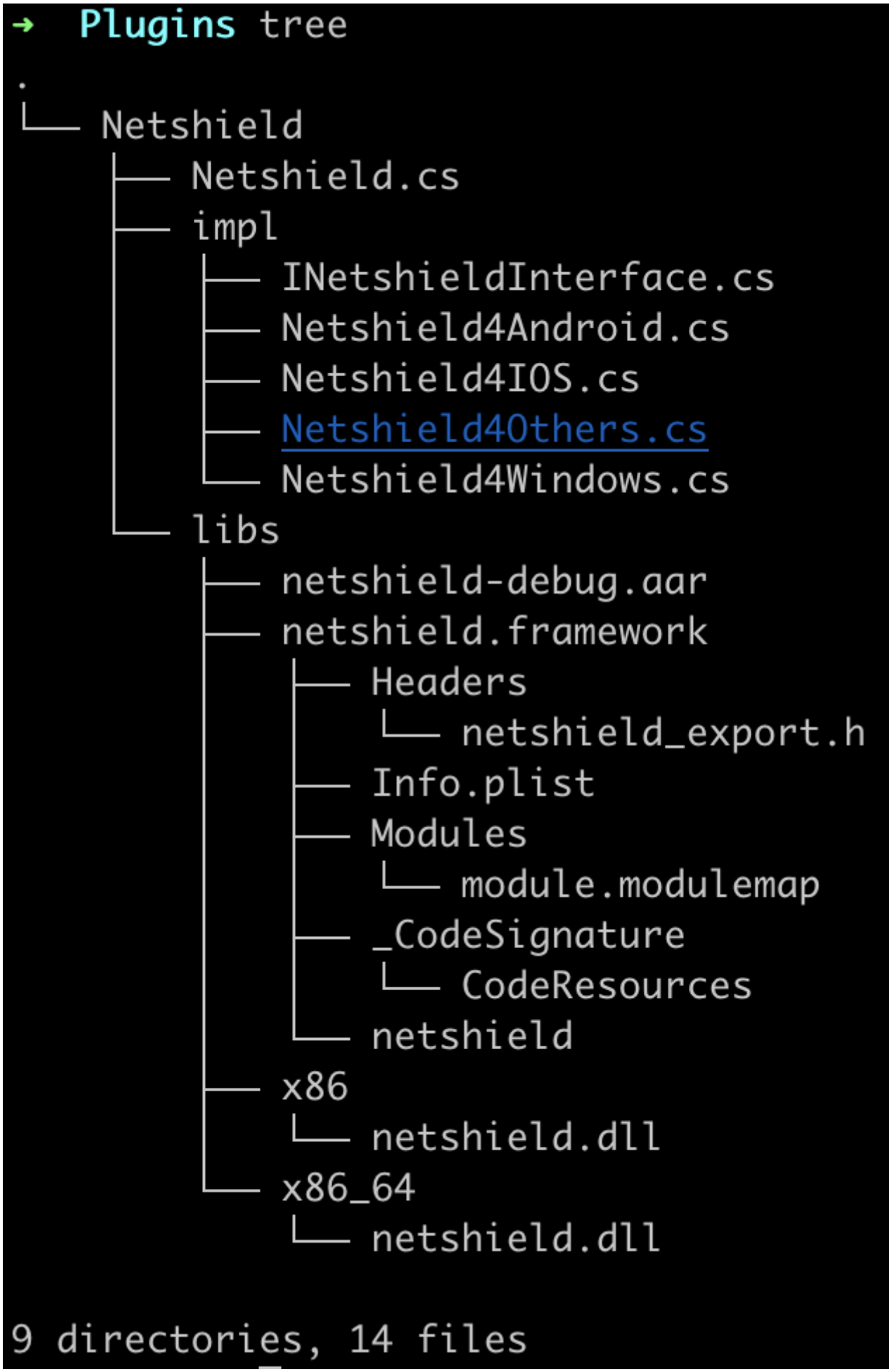
```
    pNetshield = LoadLibraryW(TEXT("netshield.dll"));
#else
    pNetshield = LoadLibraryA(TEXT("netshield.dll"));
#endif

    BOOL (*StartNetshieldService)()  = nullptr;
    const char* (*GetRealClientIP)() = nullptr;
    if (NULL != pNetshield) {
        char errmsg[512]      = {0};
        StartNetshieldService = (BOOL(*)())GetProcAddress(pNetshield,
"StartNetshieldService");
        GetRealClientIP       = (const char* (*)
())GetProcAddress(pNetshield, "GetRealClientIP");
    } else {
        MessageBoxA(NULL, "netshield.dll加载失败，请联系开发人员。",
"netshield", MB_OK | MB_ICONINFORMATION);
        exit(0);
    }
    if (nullptr != StartNetshieldService) {
        fprintf(stdout, "StartNetshieldService:%p\n",
(void*)StartNetshieldService);
        StartNetshieldService();
    }
    // 业务代码
    return 0;
}
```

## Unity

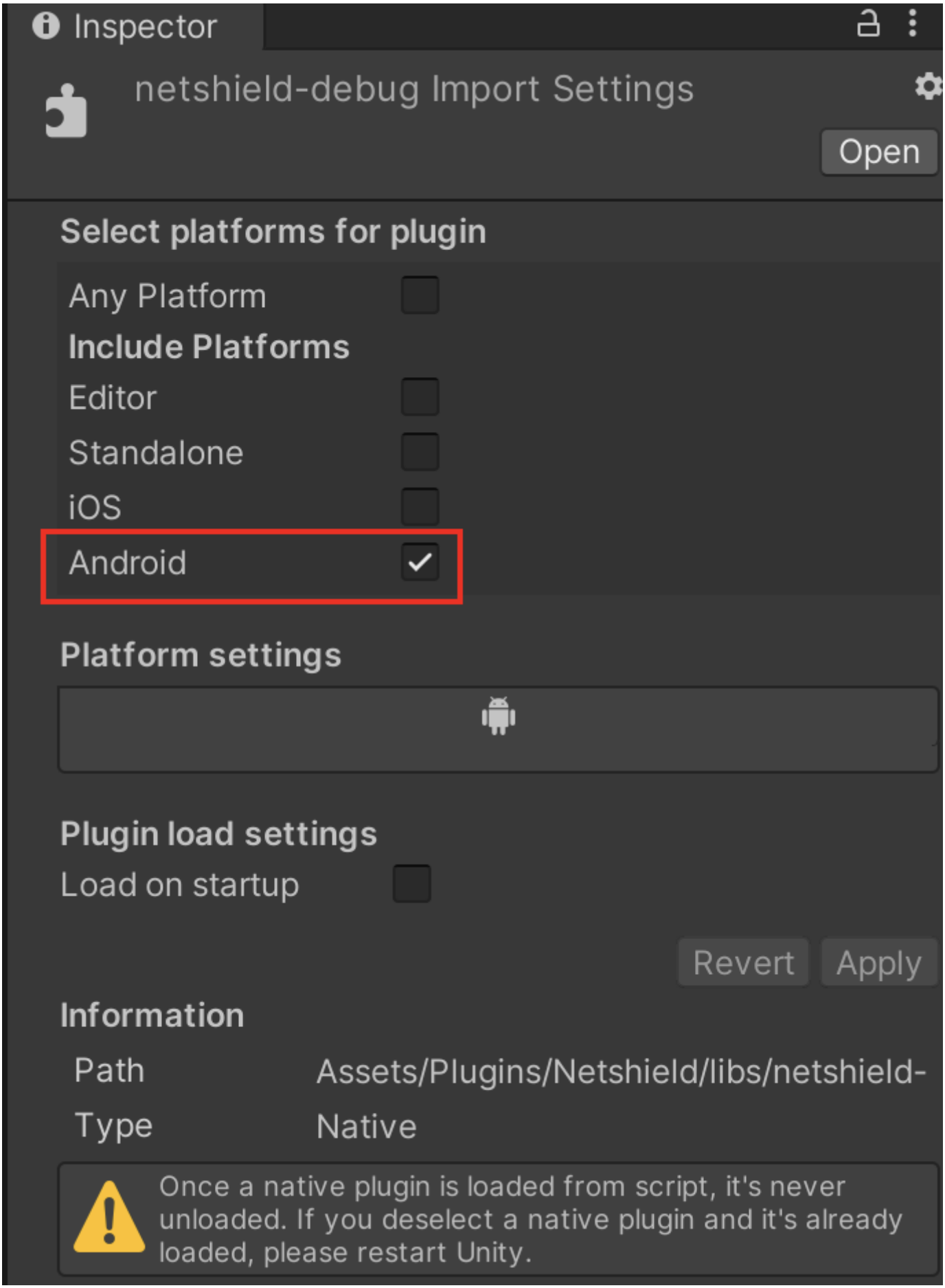
### 代码结构创建

1. 在Assets/Plugins目录下新增Netshield目录
2. 在Netshield目录下新增impl和libs目录
3. 在libs目录下新建x86和x86_64目录
4. 拷贝Netshield.cs至Assets/Plugins/Netshield目录下
5. 拷贝INetshieldInterface.cs,Netshield4IOS.cs,Netshield4Android.cs ,Netshield4Windows.cs,Netshield4Others.cs至Assets/Plugins/Netshield/impl目录下
6. 拷贝netshield-release.aar,netshield.framework至libs目录下；拷贝32位netshield.dll至x86目录下；拷贝64位netshield.dll至x86_64目录下

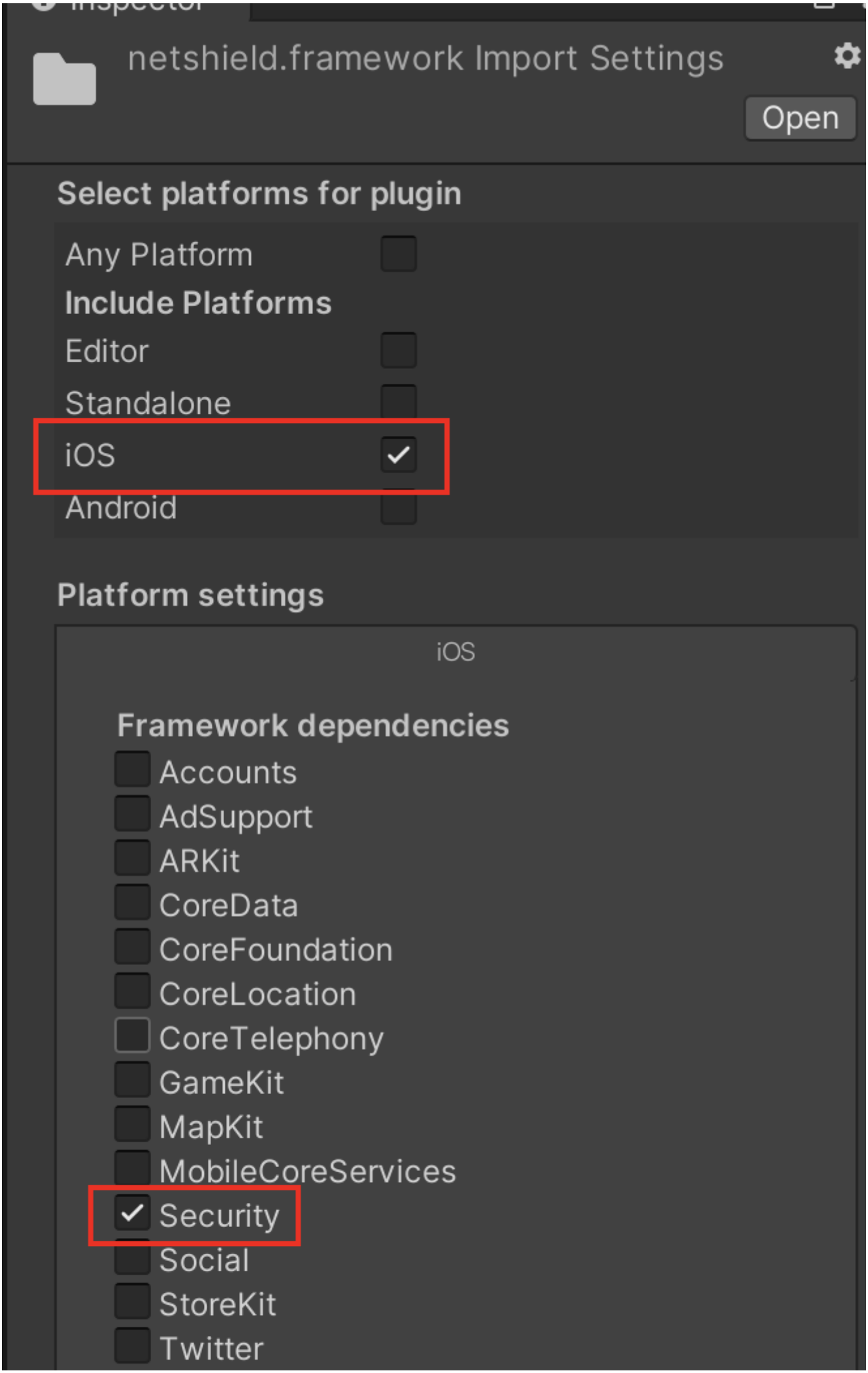

→ Plugins tree
.
└── Netshield
    ├── Netshield.cs
    ├── impl
    │   ├── INetshieldInterface.cs
    │   ├── Netshield4Android.cs
    │   ├── Netshield4IOS.cs
    │   ├── Netshield4Others.cs
    │   └── Netshield4Windows.cs
    └── libs
        ├── netshield-debug.aar
        ├── netshield.framework
        │   ├── Headers
        │   │   └── netshield_export.h
        │   ├── Info.plist
        │   ├── Modules
        │   │   └── module.modulemap
        │   ├── _CodeSignature
        │   │   └── CodeResources
        │   └── netshield
        ├── x86
        │   └── netshield.dll
        └── x86_64
            └── netshield.dll
9 directories, 14 files

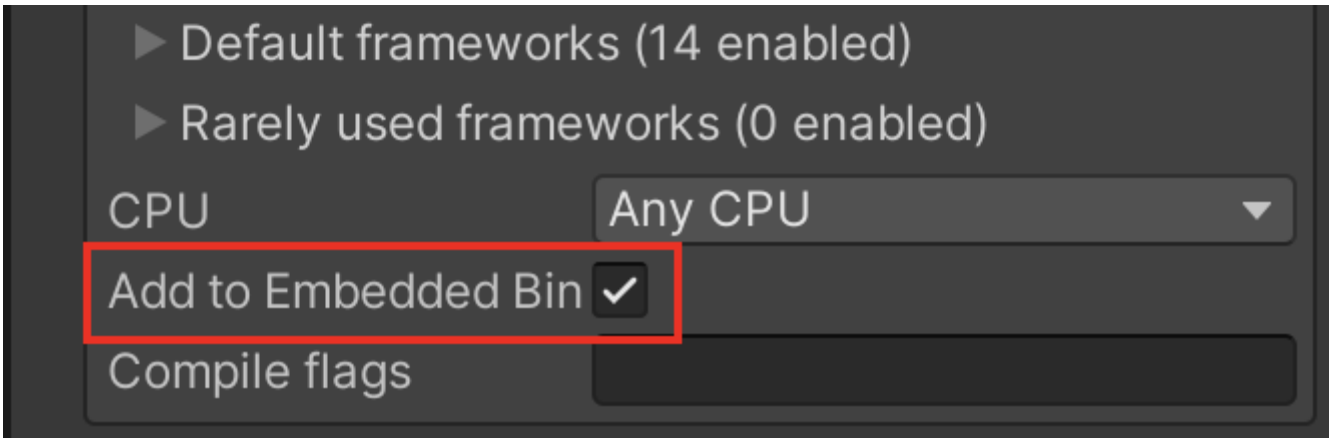
**设置SDK属性**

netshield-release.aar

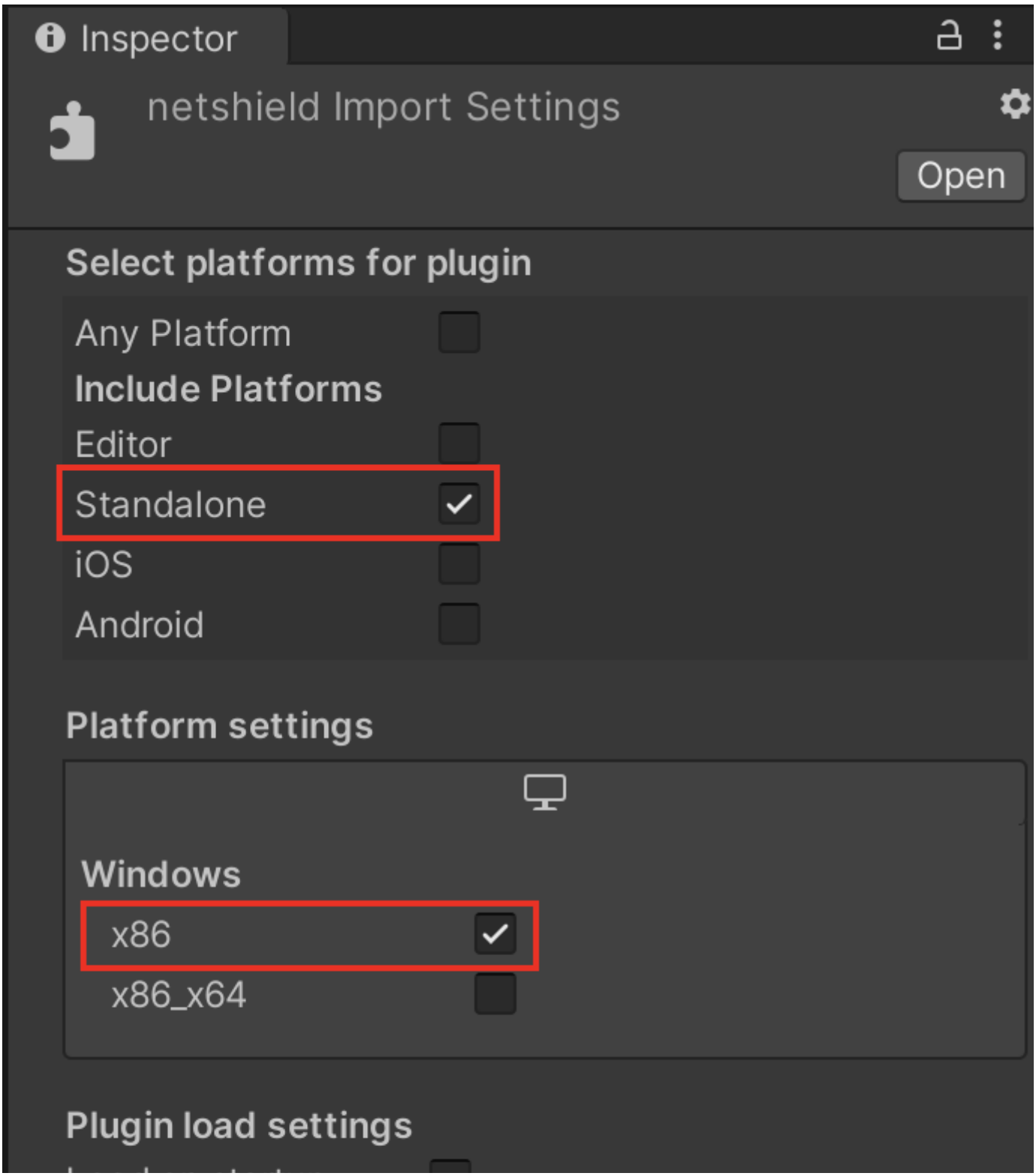


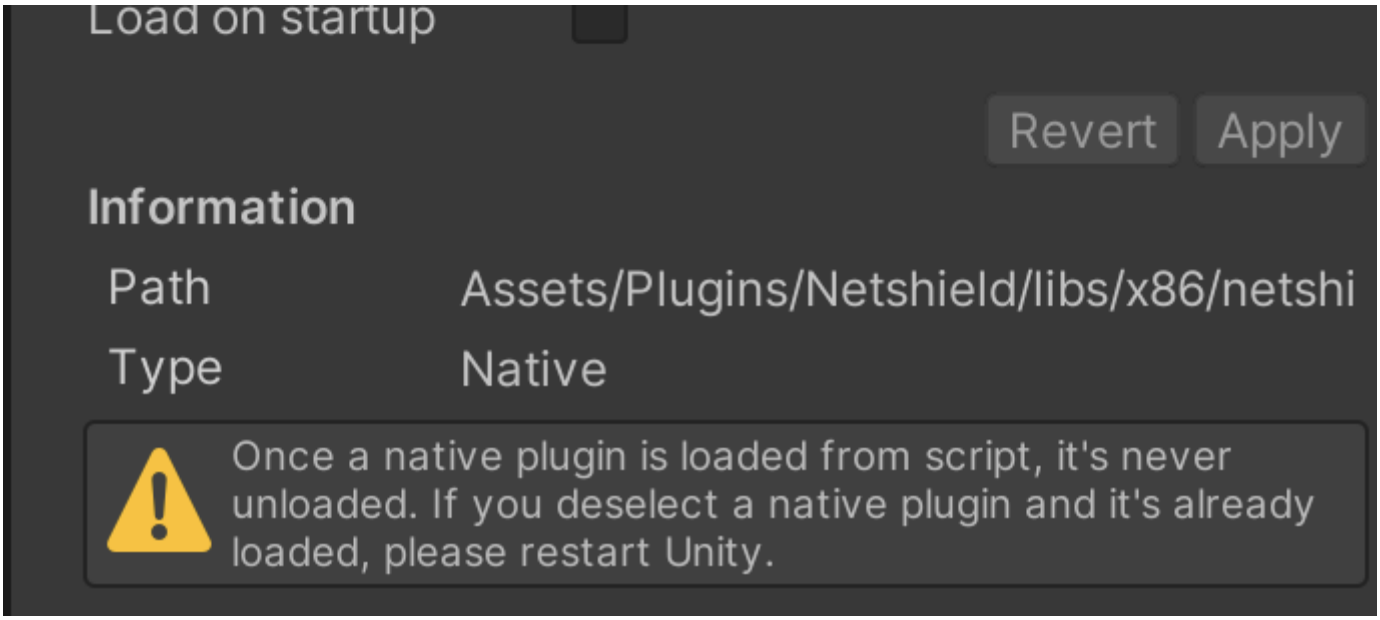

netshield.framework


netshield.framework Import Settings
Open
Select platforms for plugin
Any Platform
Include Platforms
Editor
Standalone
iOS
Android
Platform settings
iOS
Framework dependencies
Accounts
AdSupport
ARKit
CoreData
CoreFoundation
CoreLocation
CoreTelephony
GameKit
MapKit
MobileCoreServices
Security
Social
StoreKit
Twitter

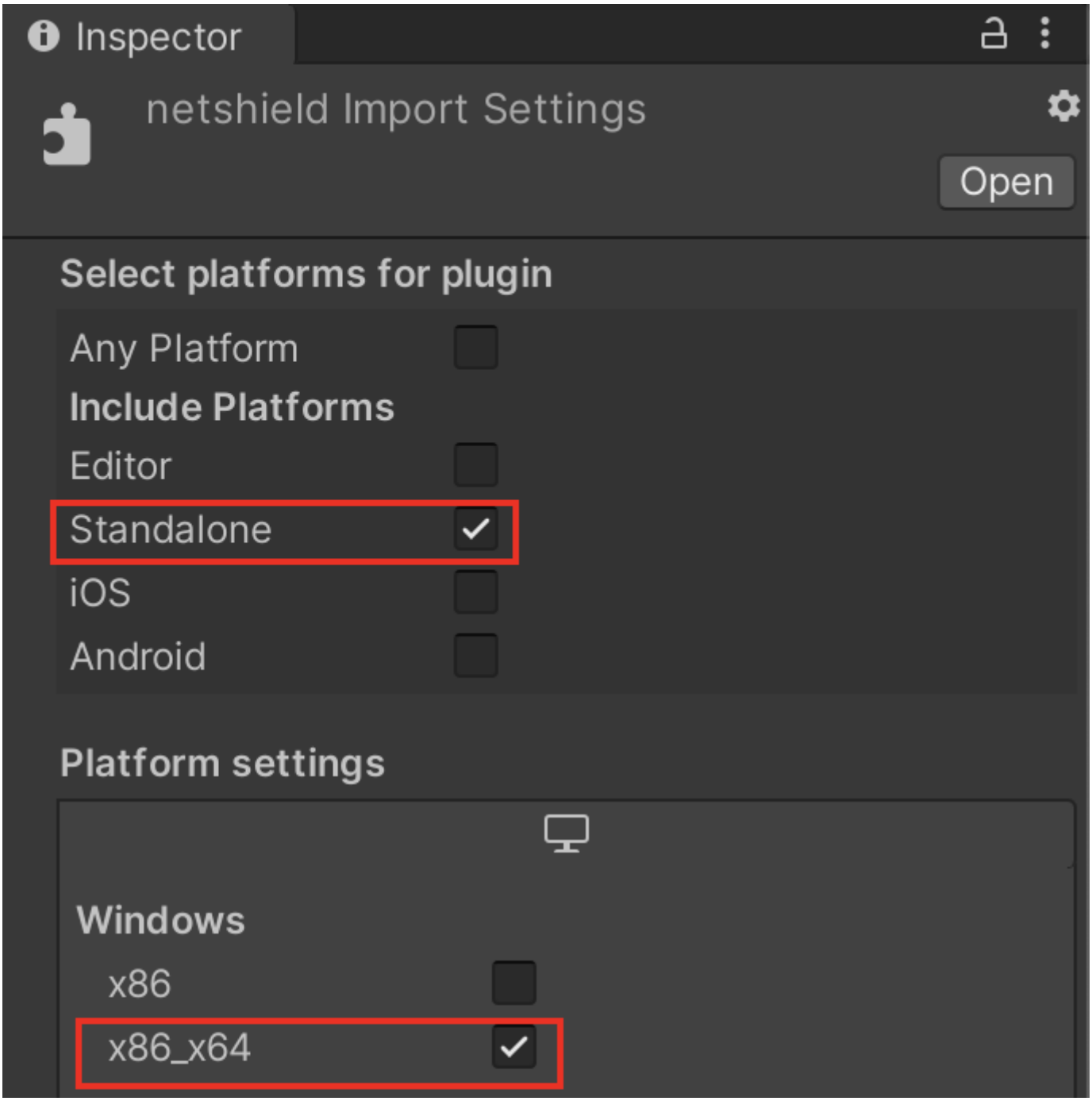


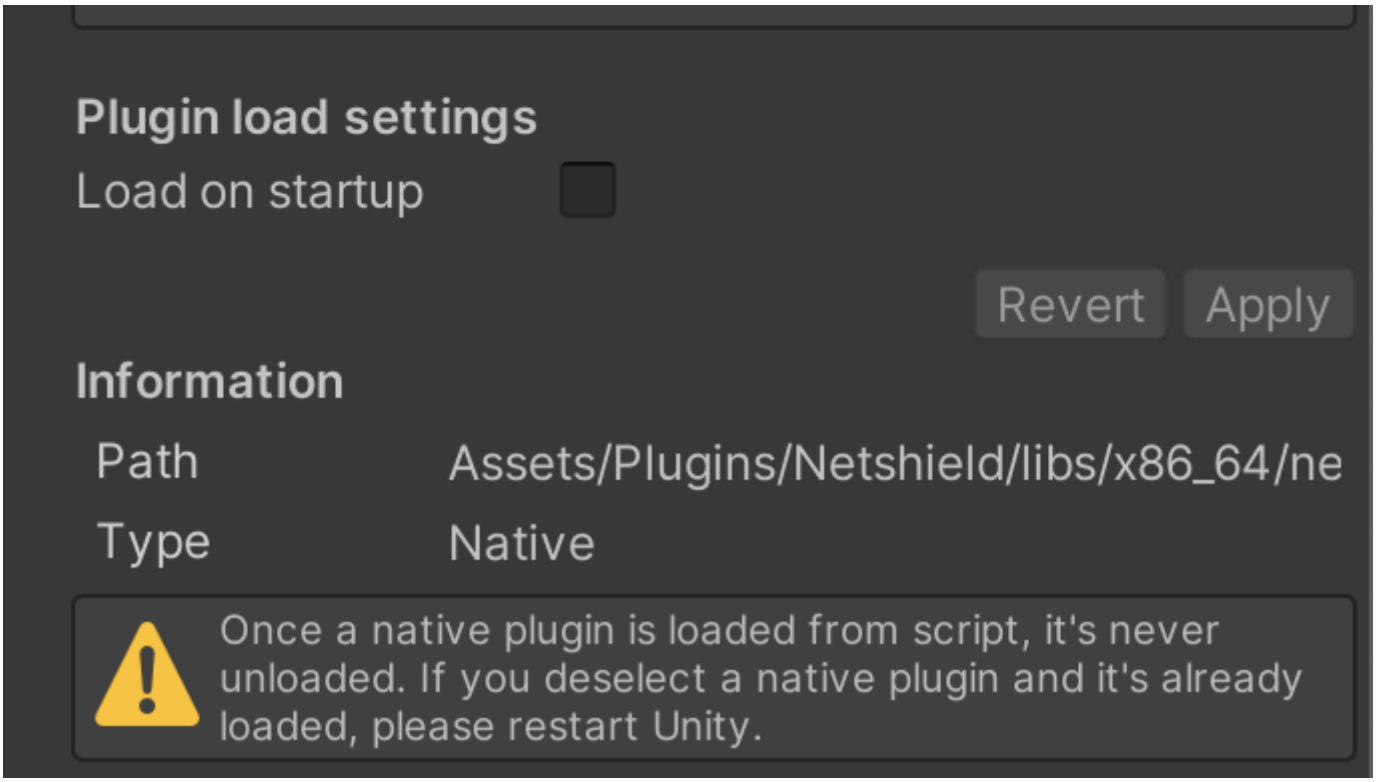

x86/netshield.dll






x86_64/netshield.dll






## 代码调用

### AndroidManifest.xml中添加以下权限

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission
android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
```

在Scene中进行调用（多个Scene需要每个Scene都调用），下列例子为绑定到Main Camera的脚本HelloWorld.cs

```
public class HelloWorld : MonoBehaviour
{
    // 应用启动
    public void Start()
    {
        // 初始化SDK
        Netshield.Init();
        // 获取真实客户端IP
        Debug.Log(Netshield.GetClientIP());
    }

    // 应用退出
    public void OnApplicationQuit()
    {
        // 释放SDK资源，在函数最后调用
```

```
        Netshield.Destroy();
    }

    // 应用焦点回调函数
    public void OnApplicationFocus(bool hasFocus)
    {
        if (hasFocus)
        {
            Netshield.Resume();
        }
        else
        {
            Netshield.Suspend();
        }
    }
}
```

## Unity相关代码

Netshield.zip

## 备注

### 问题一：

unity 2018.4.30f1 在Leaks下闪退，需要添加GameController.framework