在 Eclipse 上打包并使用 Proguard 工具混淆 jar 包

分类：*AndroidJava*

（2550）（8）

最近因为工作需要，学习到了 Android jar 包的打包与混淆。之前认为还是很简单的，但是自己深入研究下，发现还是有一些东西需要注意的，而且自己也踩了一些坑，在这里写下供同僚们借鉴借鉴。

转载请注明：

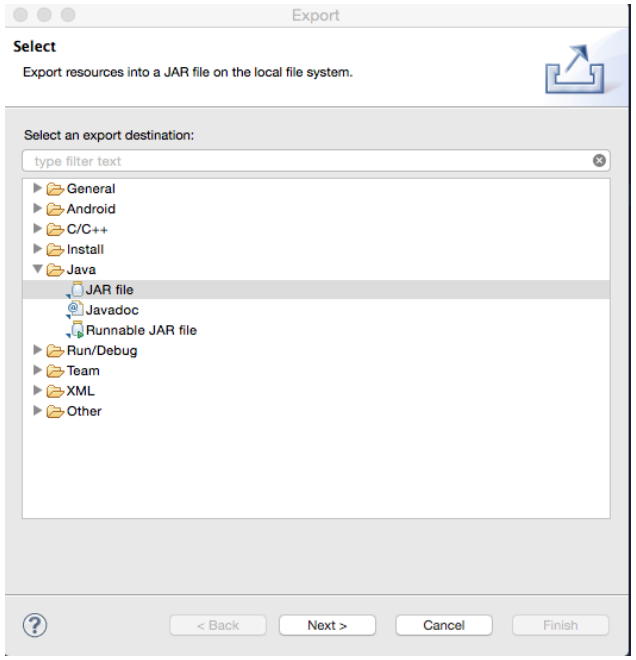
http://blog.csdn.net/aloh_a/article/details/50942751

如何打包

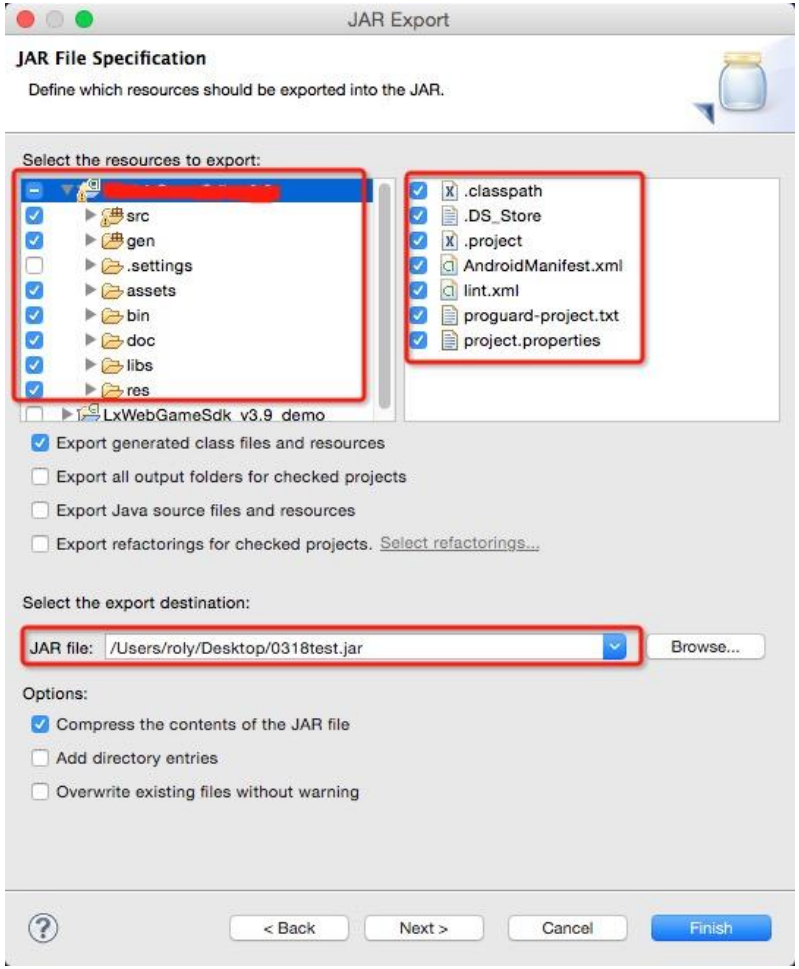
这里我以 eclipse 打包为示例，如果有朋友是用 Andorid Studio 开发的话。恕我有点 out 了。

1. 选择你的项目，右键–>Export

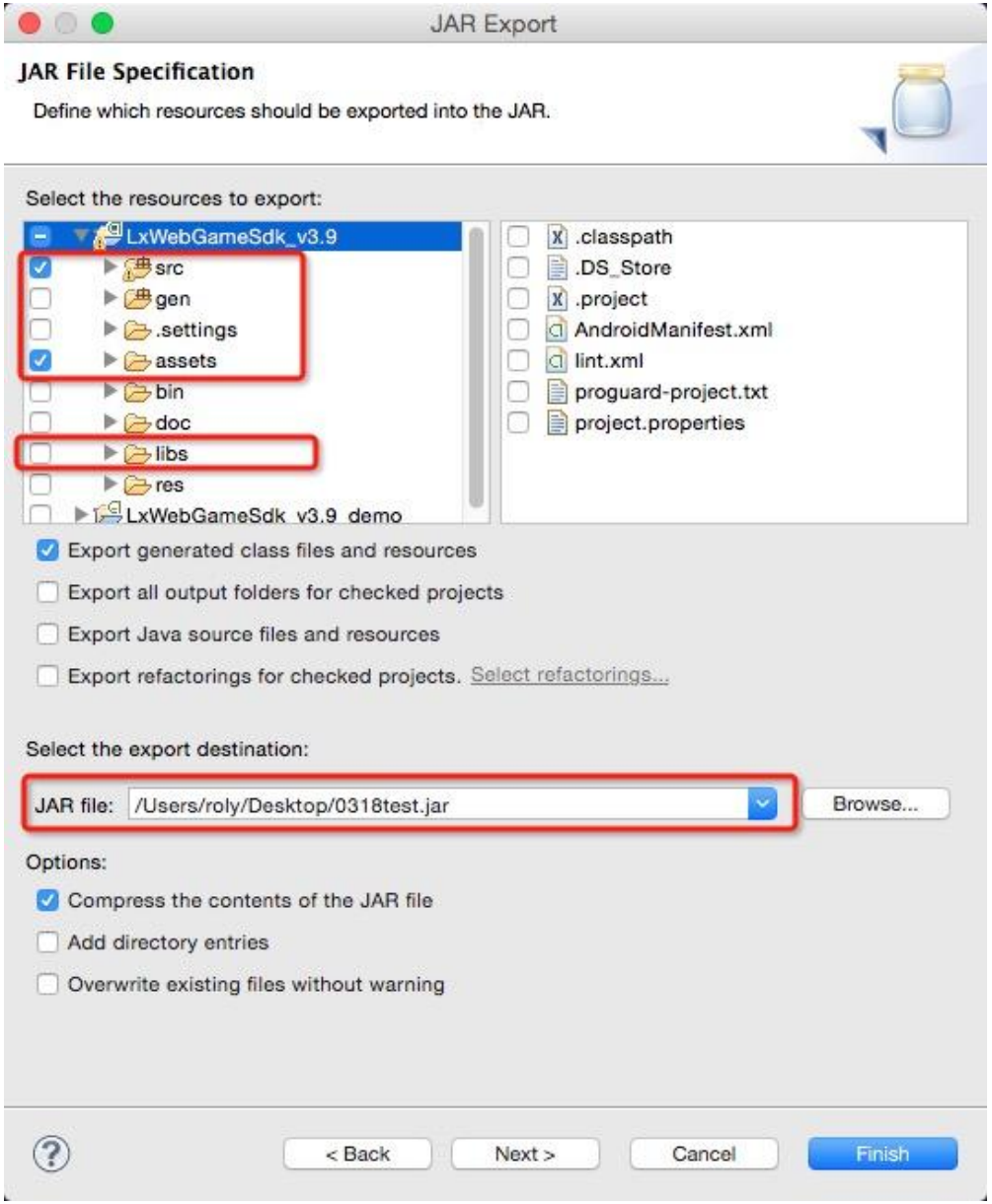
2. 选择 JAVA 分类项–>JAR file –> Next



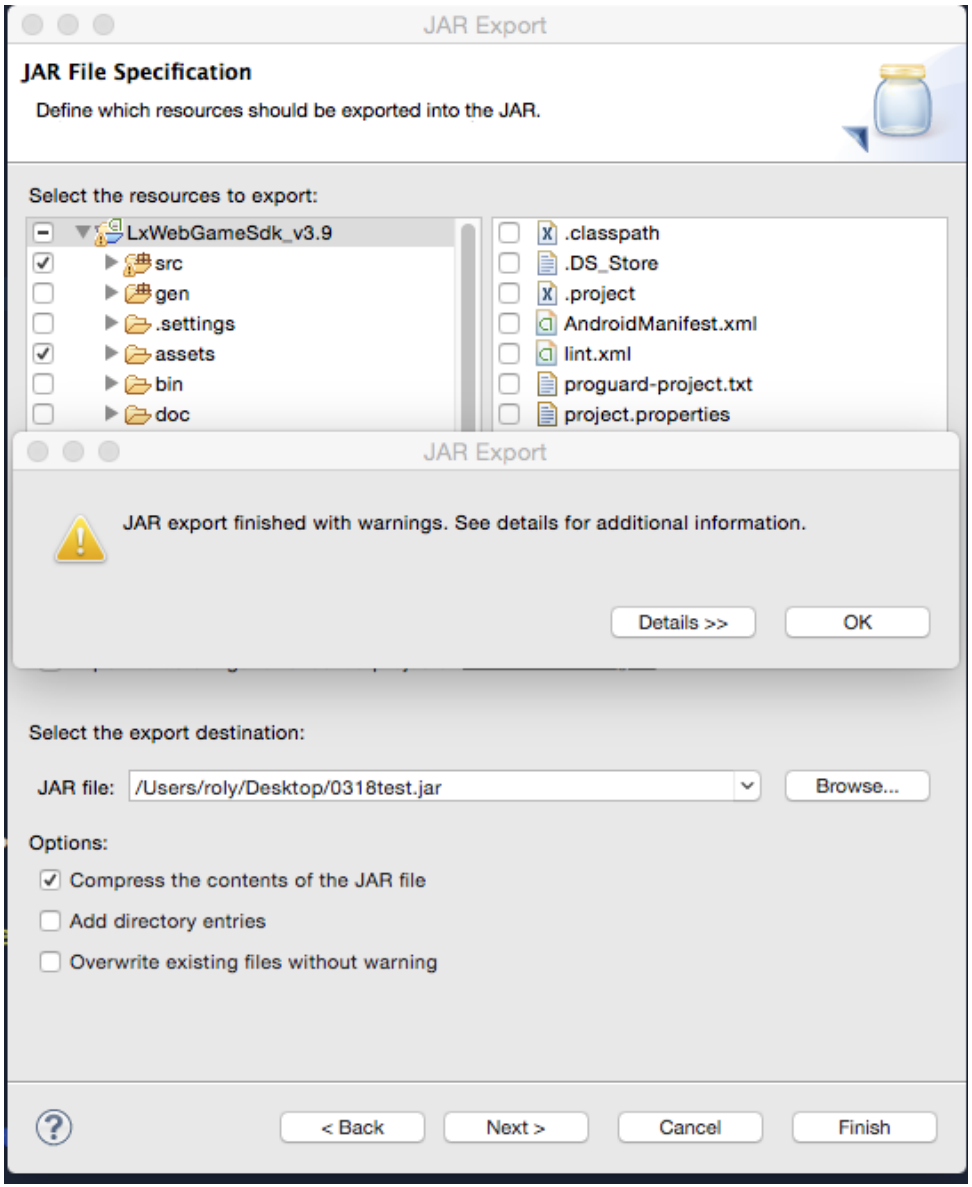
3. 这里我们看到的是 jar 包生成选项。这里我所需要的是生成第三方 sdk jar 包供他人使用，所以我这里只需要生成 src 目录下的编译好的 class 文件即可。



如图，通常 res 文件夹是不一并打包的，提供给第三方的时候，会相应地把 res 资源给到对方，并导入到对方的项目工程中。如果你的 assets 文件夹下有资源，需要勾选上一并打包。至于 libs 库，可选也可不选。选择的话，jar 大小会大一点，而这里我是没有勾选的，因为我之后会将所有 jar 包都统一合并为一个 jar 包，这样也会方便第三方的接入，不用那么麻烦地去导入多个 jar 包。最后选择我们的 jar 生成路径即可。


JAR Export
JAR File Specification
Define which resources should be exported into the JAR.
Select the resources to export:
LxWebGameSdk_v3.9
src
gen
.settings
assets
bin
doc
libs
res
LxWebGameSdk_v3.9_demo
.classpath
.DS_Store
.project
AndroidManifest.xml
lint.xml
proguard-project.txt
project.properties
Export generated class files and resources
Export all output folders for checked projects
Export Java source files and resources
Export refactorings for checked projects. Select refactorings...
Select the export destination:
JAR file: /Users/roly/Desktop/0318test.jar
Browse...
Options:
Compress the contents of the JAR file
Add directory entries
Overwrite existing files without warning
< Back
Next >
Cancel
Finish

4. 最后选择好选项，我们点击 finish 按钮，看到如下图所示，那么我们的 jar 包就生成成功了。



混淆

在做混淆之前，我们需要了解，为什么要做混淆。这里我参考了下郭霖大神的博客，有兴趣了解下的朋友可以先看看。

Android 安全攻防战，反编译与混淆技术完全解析

http://blog.csdn.net/guolin_blog/article/details/49738023

http://blog.csdn.net/guolin_blog/article/details/50451259

郭神的博客我简要概述下就是，我们生成的 jar 包以及 apk 文件其实是并不安全的(具体可参考未做加密的潜蜻蜓 FM 事件)，通过反编译方面的知识，包括反编译代码、反编译资源，我们可以得到开发者的源码，甚至是重新打包，篡改开发者的文件而达到某种目的。

使用 Proguard 工具混淆 jar 包

谷歌非常人性化地为我们提供了一个混淆工具，我们可以在 sdk->tools->proguard->bin->proguardgui 路径下找到它，打开如下图。


ProGuard
ProGuard
Input/Output
Shrinking
Obfuscation
Optimization
Information
Process
ReTrace
ProGuard
Shrinking
Optimization
ProGuard
Obfuscation
Preverification
Developed by Eric Lafortune
Welcome to ProGuard, version 4.7
ProGuard is a free class file shrinker, optimizer, obfuscator, and preverifier.
With this GUI, you can create, load, modify, and save ProGuard configurations.
You can then process your code right away, or you can run ProGuard from the command line using your saved configuration.
With the ReTrace part of this GUI you can de-obfuscate your stack traces.
ProGuard and ReTrace are written and maintained by Eric Lafortune.
Official site at Sourceforge: http://proguard.sourceforge.net/
Professional support by Saikoa: http://www.saikoa.com/
Distributed under the GNU General Public License.
Copyright © 2002-2011.
Load configuration...
Next

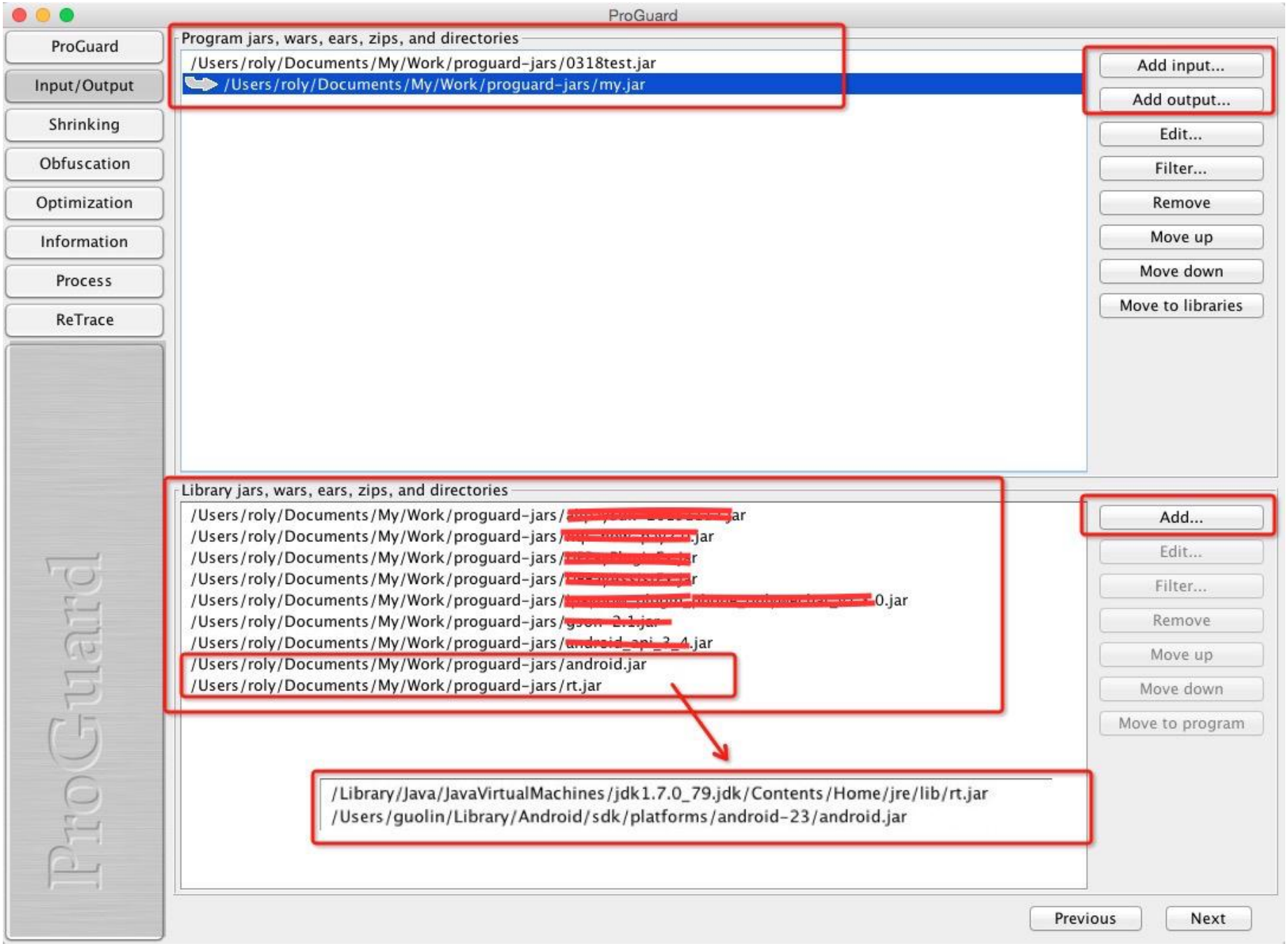
这里便是 proguard 混淆工具，看到右下角有一个 Load configuration 按钮，其实它的本质还是通过 proguard-android 文件来进行配置混淆选项的，proguard-android 文件我们可以在 sdk–>tools–>proguard–>proguard-android.txt 下找到它，通过配置，我们便可以拥有自己的一份混淆配置文件了。

接下来我们点击左栏的 Input/Output 按钮，进入下一步。

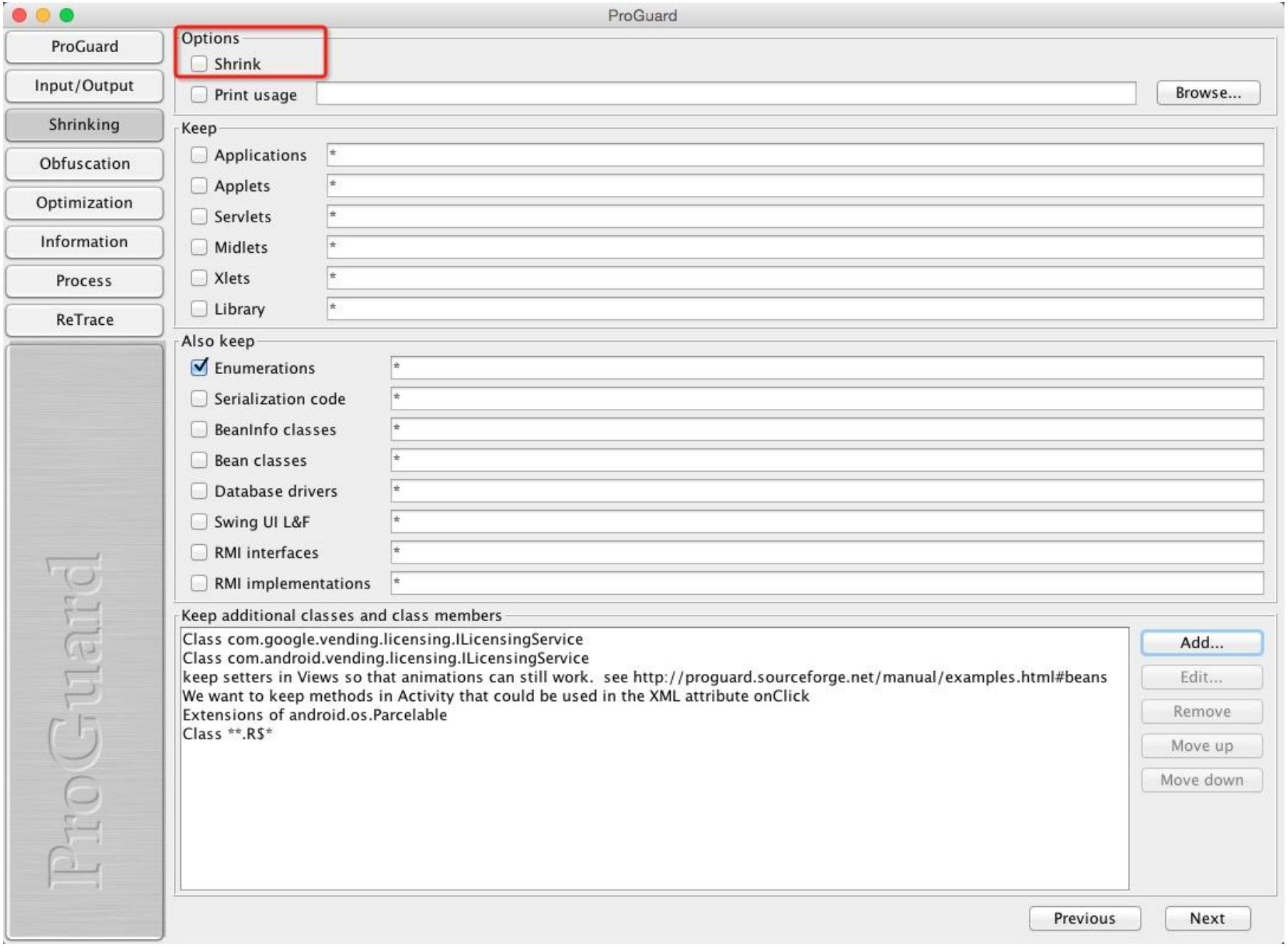
这里我们看到上栏，是配置将要混淆的 jar 文件以及混淆之后的文件生成路径，我们可以在右栏进行配置。

下栏是我们要进行混淆的工程生成的 jar 文件的其他依赖 jar 包。

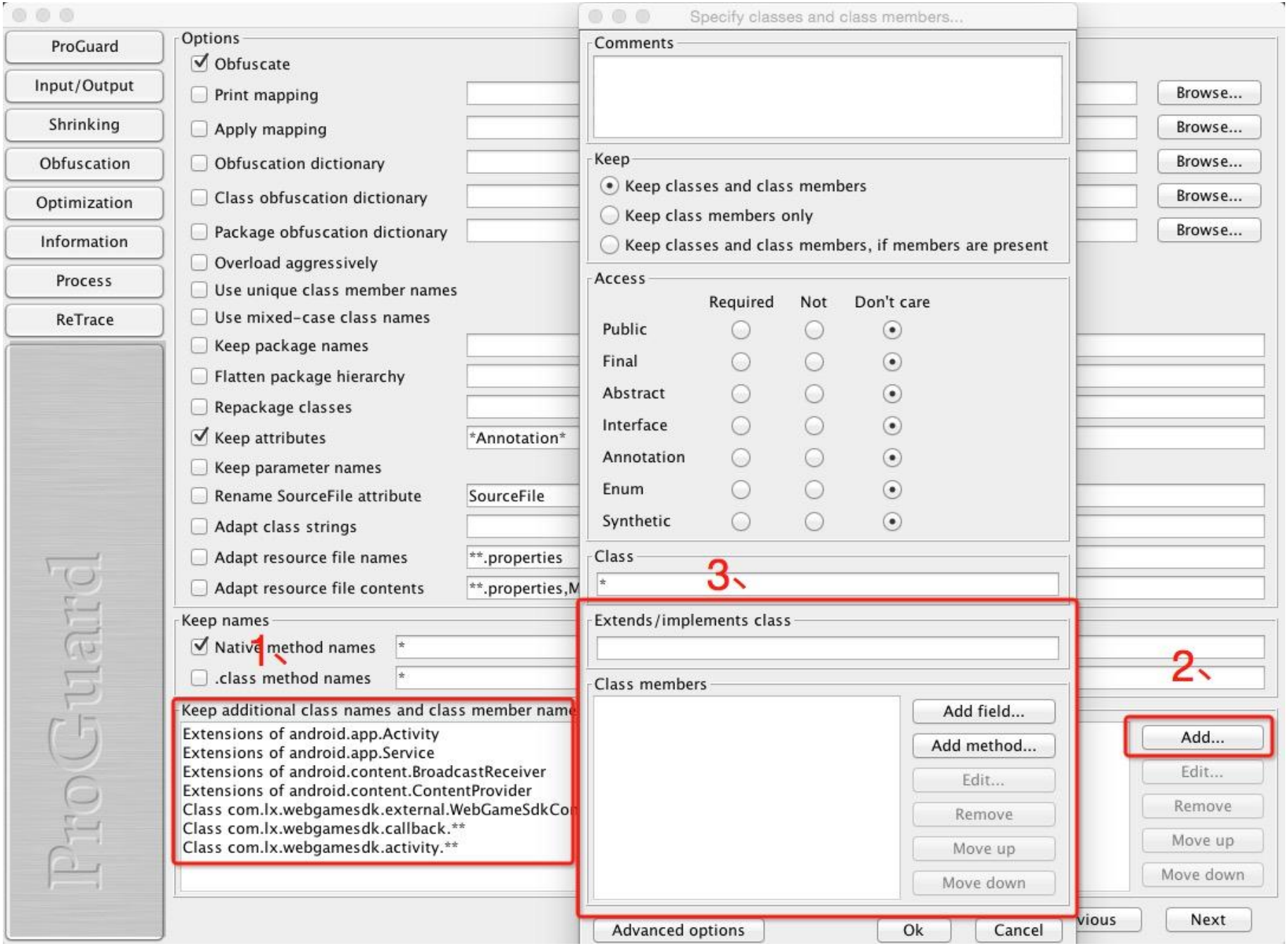
路径都是我本机的，我们看到最后 2 个 jar 包比较关键。第一个是你工程开发 api 的 android.jar 包，路径如图，第二个 java 的 rt.jar 包，路径依如图。这里我坑了一下，因为我自己安装的 1.8 的 jdk，但是我使用我本机的 rt.jar 包却混淆失败了，会报错如下：ProGuard says Unsupported class version number [52.0] (maximum 51.0, Java 1.7) with sbt-proguard。原因在于 proguard 只能支持最高 1.7 版本的 jdk， 所以这里我就坑了，之后是找朋友拿了 1.7 版本的该 jar 包，最后才混淆成功。


ProGuard
ProGuard
Input/Output
Shrinking
Obfuscation
Optimization
Information
Process
ReTrace
ProGuard
Program jars, wars, ears, zips, and directories
/Users/roly/Documents/My/Work/proguard-jars/0318test.jar
/Users/roly/Documents/My/Work/proguard-jars/my.jar
Add input...
Add output...
Edit...
Filter...
Remove
Move up
Move down
Move to libraries
Library jars, wars, ears, zips, and directories
/Users/roly/Documents/My/Work/proguard-jars/
/Users/roly/Documents/My/Work/proguard-jars/
/Users/roly/Documents/My/Work/proguard-jars/
/Users/roly/Documents/My/Work/proguard-jars/
/Users/roly/Documents/My/Work/proguard-jars/
/Users/roly/Documents/My/Work/proguard-jars/
/Users/roly/Documents/My/Work/proguard-jars/
/Users/roly/Documents/My/Work/proguard-jars/android.jar
/Users/roly/Documents/My/Work/proguard-jars/rt.jar
/Library/Java/JavaVirtualMachines/jdk1.7.0_79.jdk/Contents/Home/jre/lib/rt.jar
/Users/guolin/Library/Android/sdk/platforms/android-23/android.jar
Add...
Edit...
Filter...
Remove
Move up
Move down
Move to program
Previous
Next

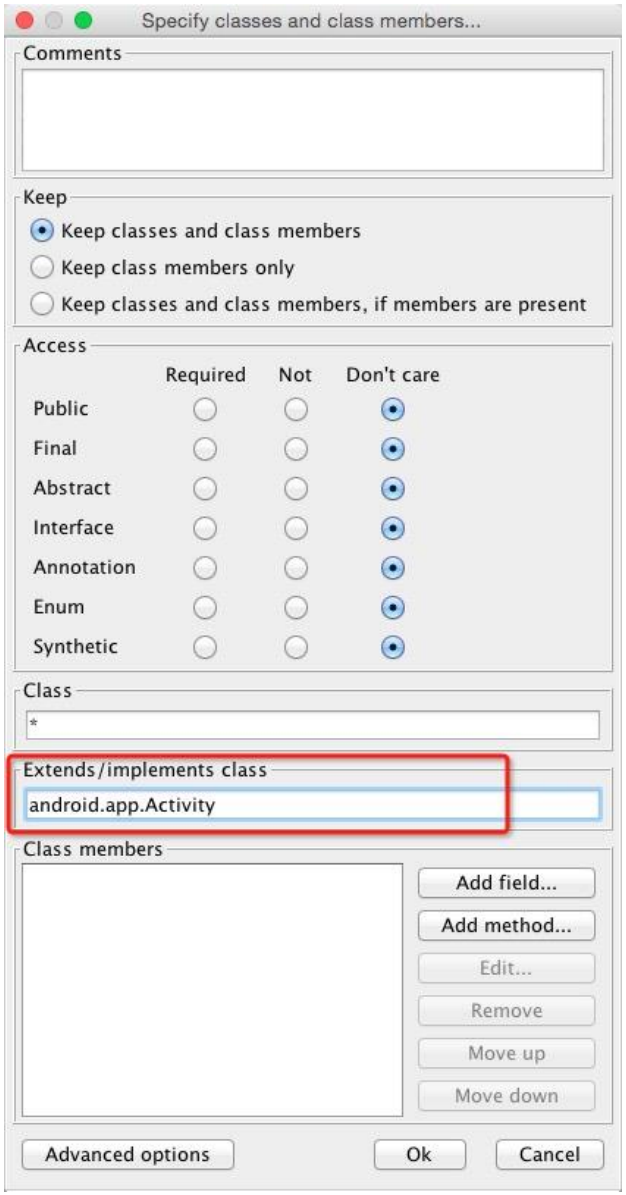
接下来我们点击 next 按钮进入下一步，进入 Shrinking 选项，记得要将 Shrink 选项钩掉，因为我们这个 Jar 包是独立存在的，没有任何项目引用，如果钩中 Shrink 选项的话就会认为我们所有的代码都是无用的，从而把所有代码全压缩掉，导出一个空的 Jar 包。


ProGuard
ProGuard
Input/Output
Shrinking
Obfuscation
Optimization
Information
Process
ReTrace
ProGuard
Options
Shrink
Print usage
Browse...
Keep
Applications
Applets
Servlets
Midlets
Xlets
Library
Also keep
Enumerations
Serialization code
BeanInfo classes
Bean classes
Database drivers
Swing UI L&F
RMI interfaces
RMI implementations
Keep additional classes and class members
Class com.google.vending.licensing.ILicensingService
Class com.android.vending.licensing.ILicensingService
keep setters in Views so that animations can still work. see http://proguard.sourceforge.net/manual/examples.html#beans
We want to keep methods in Activity that could be used in the XML attribute onClick
Extensions of android.os.Parcelable
Class **.R$*
Add...
Edit...
Remove
Move up
Move down
Previous
Next

继续点击 Next 进入 Obfuscation 界面，在这里可以添加一些混淆的逻辑，和混淆 APK 时不同的是，这里并不会自动帮我们排除混淆四大组件，因此必须要手动声明一下才行，以及我们可以添加我们一些自定义不混淆的类或者变量。如 1 所示，即我所自定义的混淆规则，2 即是 add 按钮，点击它，将出现 3 界面，我们即可在其上编写排除逻辑。

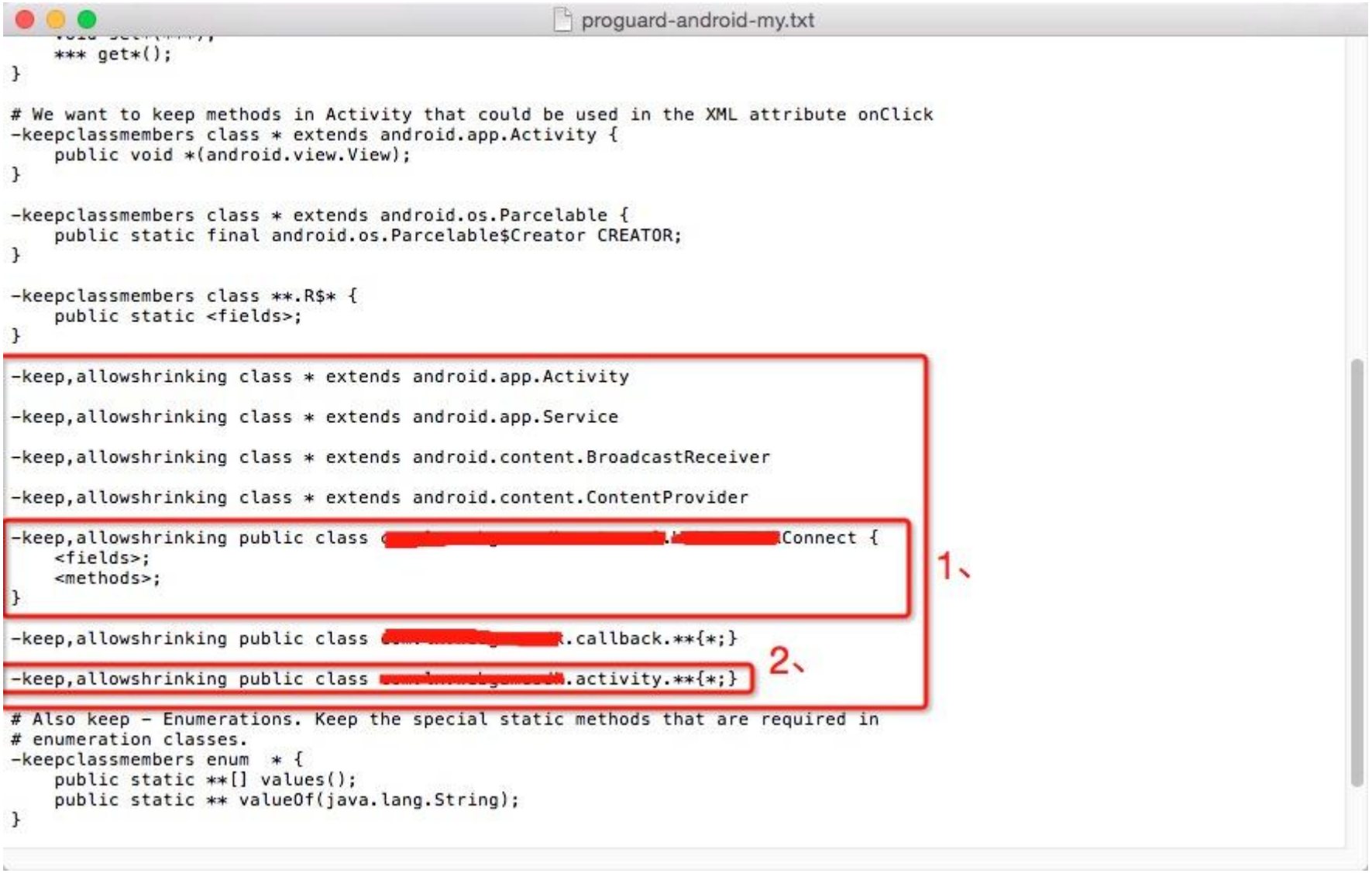

ProGuard
Input/Output
Shrinking
Obfuscation
Optimization
Information
Process
ReTrace
ProGuard
Options
Obfuscate
Print mapping
Apply mapping
Obfuscation dictionary
Class obfuscation dictionary
Package obfuscation dictionary
Overload aggressively
Use unique class member names
Use mixed-case class names
Keep package names
Flatten package hierarchy
Repackage classes
Keep attributes
Annotation
Keep parameter names
Rename SourceFile attribute
SourceFile
Adapt class strings
Adapt resource file names
**.properties
Adapt resource file contents
**.properties,M
Browse...
Keep names
Native method names
.class method names
1、
Keep additional class names and class member name
Extensions of android.app.Activity
Extensions of android.app.Service
Extensions of android.content.BroadcastReceiver
Extensions of android.content.ContentProvider
Class com.lx.webgamesdk.external.WebGameSdkCo
Class com.lx.webgamesdk.callback.**
Class com.lx.webgamesdk.activity.**
Specify classes and class members...
Comments
Keep
Keep classes and class members
Keep class members only
Keep classes and class members, if members are present
Access
Required
Not
Don't care
Public
Final
Abstract
Interface
Annotation
Enum
Synthetic
Class
3、
Extends/implements class
Class members
Add field...
Add method...
Edit...
Remove
Move up
Move down
2、
Add...
Edit...
Remove
Move up
Move down
Advanced options
Ok
Cancel
vious
Next

假设我们这里要混淆 Activity 类，如下所示。最后记得按 OK 保存。



下图是我最后混淆的结果，如果我们不混淆某个类的方法以及变量，即参照 1 混淆规则，如果我们不混淆某个包下的所有类的方法以及变量，参

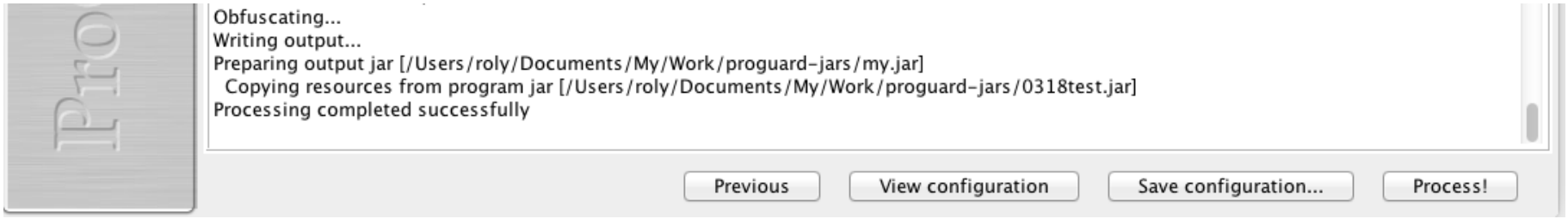
照 2 的混淆规则。



继续点击 Next 进入 Optimiazation 界面，不用修改任何东西，因为我们本身就不启用 Optimization 功能。继续点击 Next 进入 Information 界面，也不用修改任何东西，因为我们也不启用 Preverification 功能。

接着点击 Next，进入 Process 界面，在这里可以通过点击 View configuration 按钮来预览一下目前我们的混淆配置文件。我们亦可点击 Save configuration 按钮，来保存一份我们自己的混淆配置文件。

最后点击 Process！ 按钮，即可开始进行混淆了。



最后当我们看到 Success！ 就说明我们混淆成功了！如果有 warning 提示，那么即按照提示做相应的修改吧。希望这篇博文能对各位起到一点微小的帮助。