[在Eclipse上打包并使用Proguard工具混淆jar包](http://blog.csdn.net/Aloh_A/article/details/50942751)

分类：***AndroidJava***

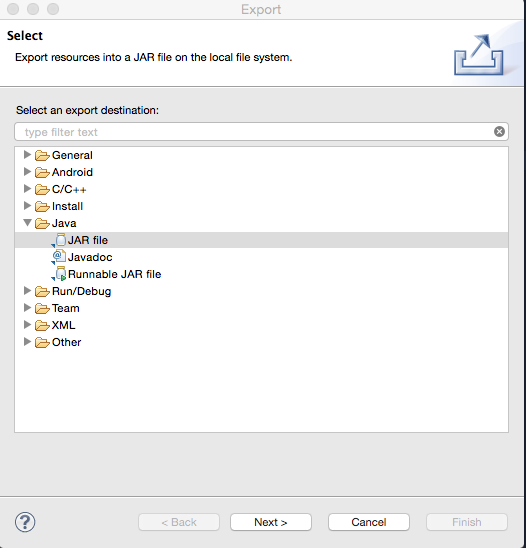
 （2550）  （8）

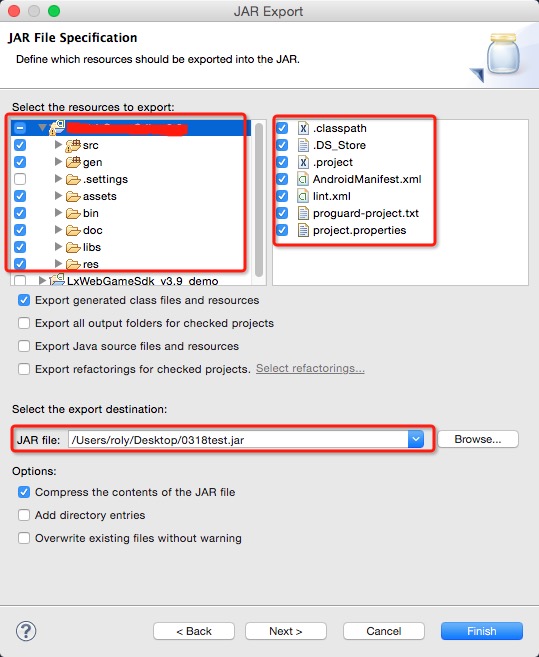
最近因为工作需要，学习到了Android jar包的打包与混淆。之前认为还是很简单的，但是自己深入研究下，发现还是有一些东西需要注意的，而且自己也踩了一些坑，在这里写下供同僚们借鉴借鉴。   
转载请注明：

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

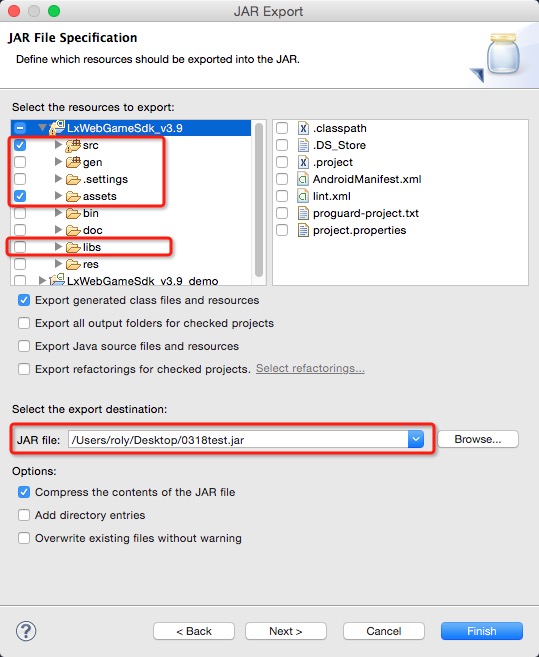
如何打包

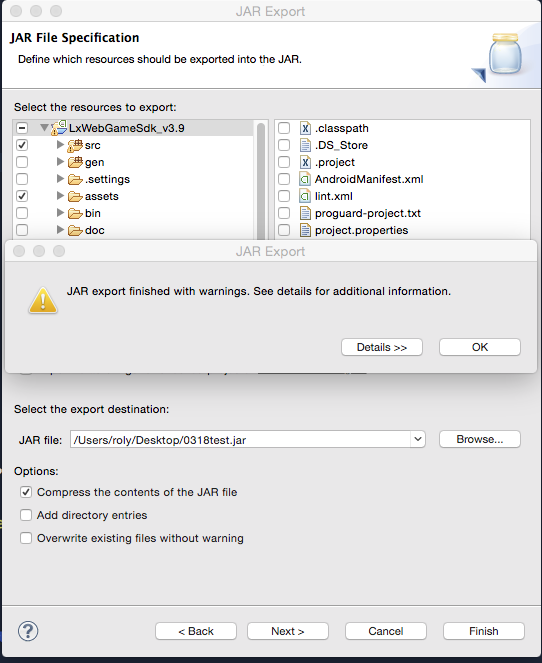
这里我以eclipse 打包为示例，如果有朋友是用Andorid Studio开发的话。恕我有点out了。

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。



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。

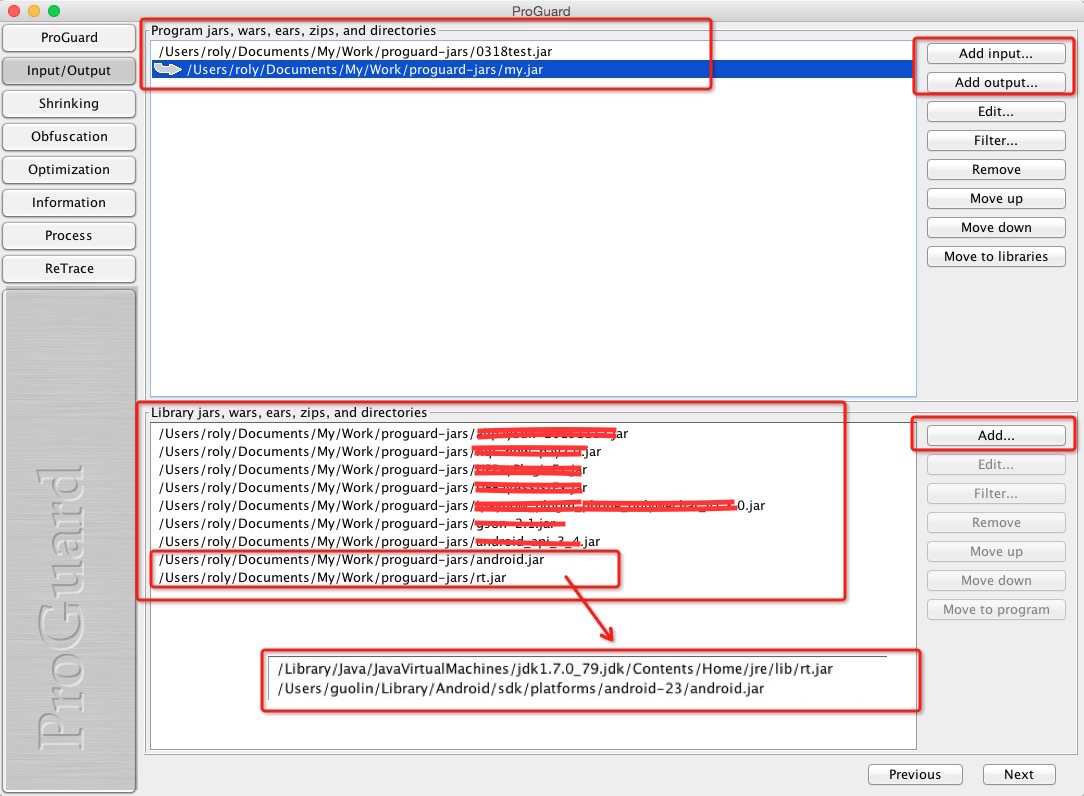
**使用Proguard工具混淆jar包**   
谷歌非常人性化地为我们提供了一个混淆工具，我们可以在sdk–>tools–>proguard–>bin–>proguardgui 路径下找到它，打开如下图。   
  
这里便是proguard混淆工具，看到右下角有一个Load configuration 按钮，其实它的本质还是通过proguard-android 文件来进行配置混淆选项的，proguard-android 文件我们可以在sdk–>tools–>proguard–>proguard-android.txt 下找到它，通过配置，我们便可以拥有自己的一份混淆配置文件了。

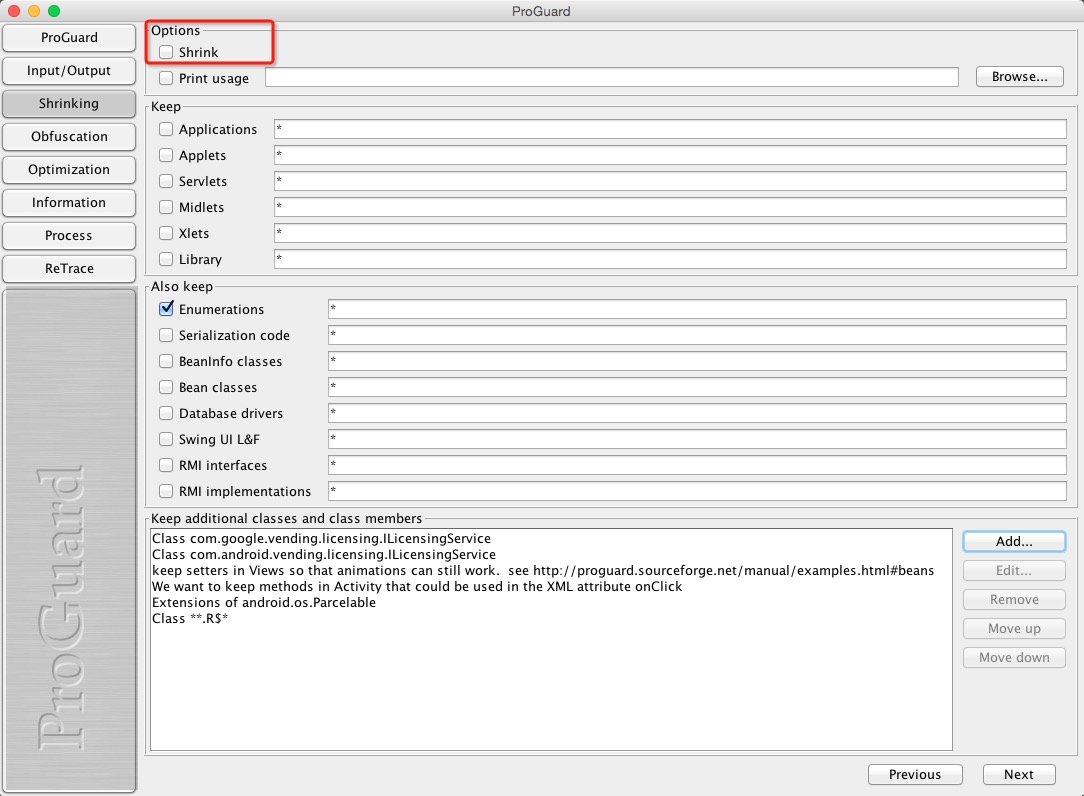
接下来我们点击左栏的Input/Output 按钮，进入下一步。

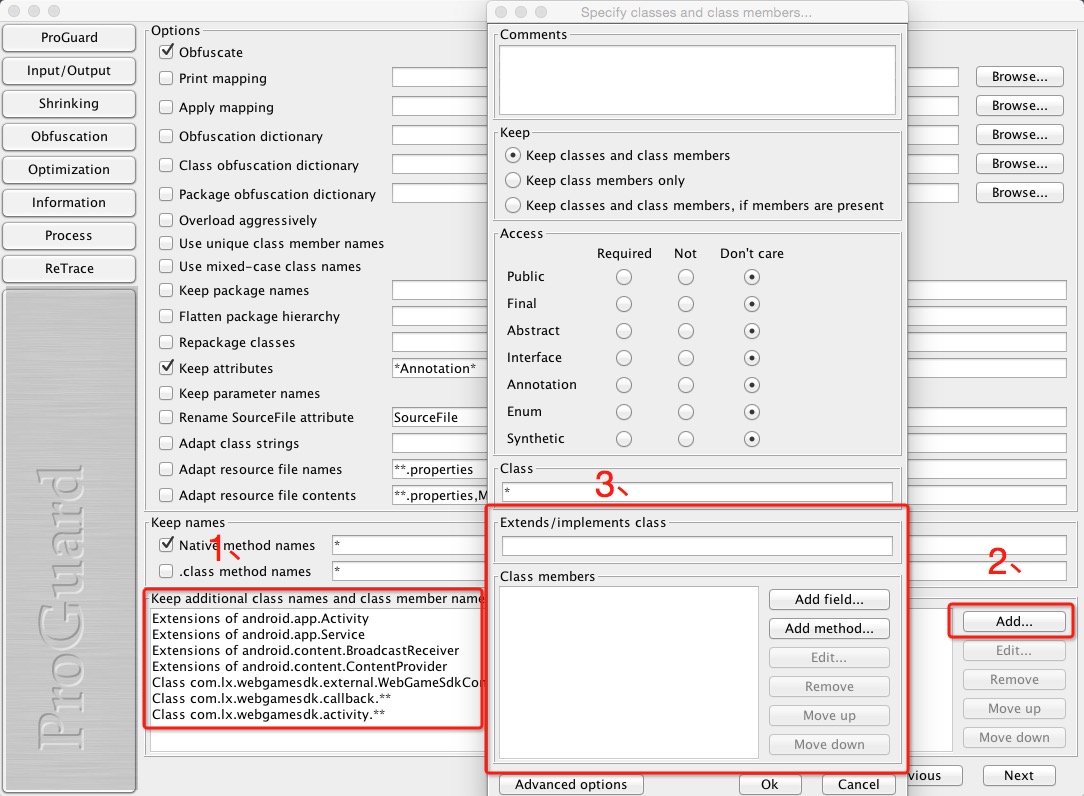
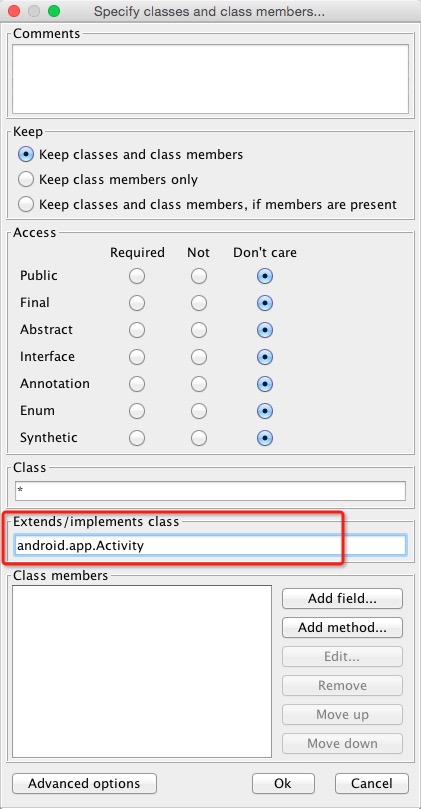
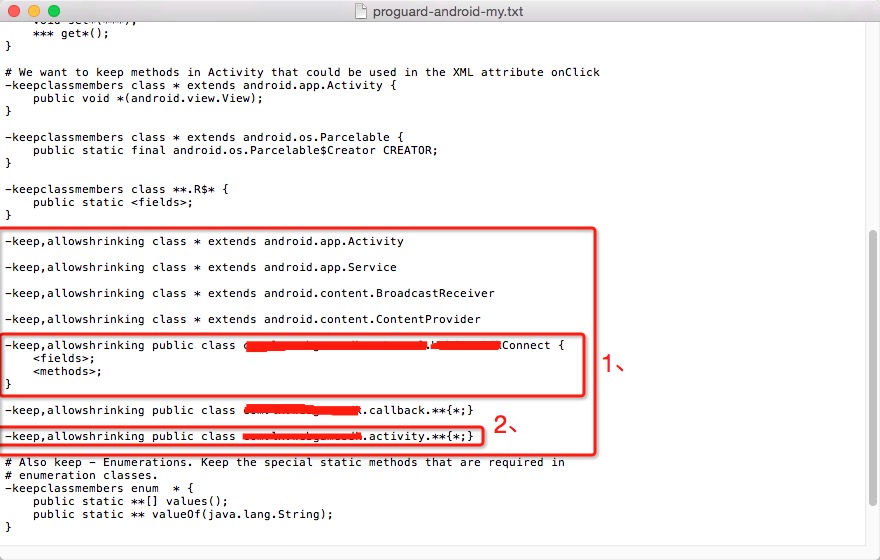
这里我们看到上栏，是配置将要混淆的jar 文件以及混淆之后的文件生成路径，我们可以在右栏进行配置。

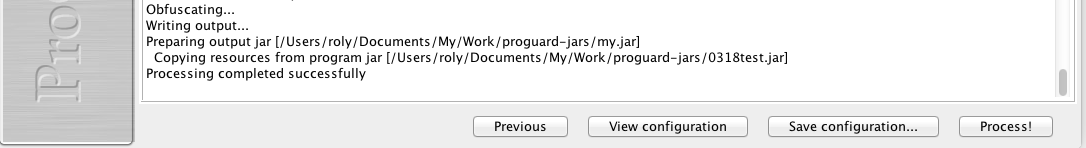
下栏是我们要进行混淆的工程生成的jar文件的其他依赖jar包。

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。



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。   


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。   
  
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。

版权声明：本文为博主原创文章，未经博主允许不得转载。