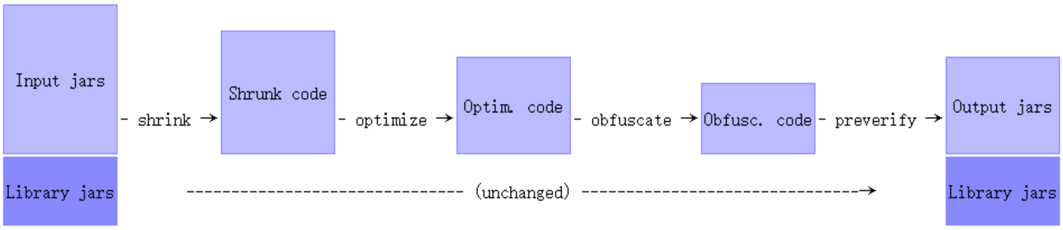
Proguard 简要语法手册



默认优化以及预校验是没有打开的

语法

-libraryjars class_path 应用的依赖包，如 android-support-v4

-keep [,modifier,...] class_specification 不混淆某些类以及成员

-keepclassmembers [,modifier,...] class_specification 不混淆类的成员

-keepclasseswithmembers [,modifier,...] class_specification 不混淆类及其成员

-keepnames class_specification 不混淆类及其成员（如果经过压缩后还存在）

-keepclassmembernames class_specification 不混淆类的成员名（同上）

-keepclasseswithmembernames class_specification 不混淆类及其成员名（同上）

-assumenosideeffects class_specification 假设调用不产生任何影响，在 proguard 代码优化时

将该调用 remove 掉。如 system.out.println 和 Log.v 等等

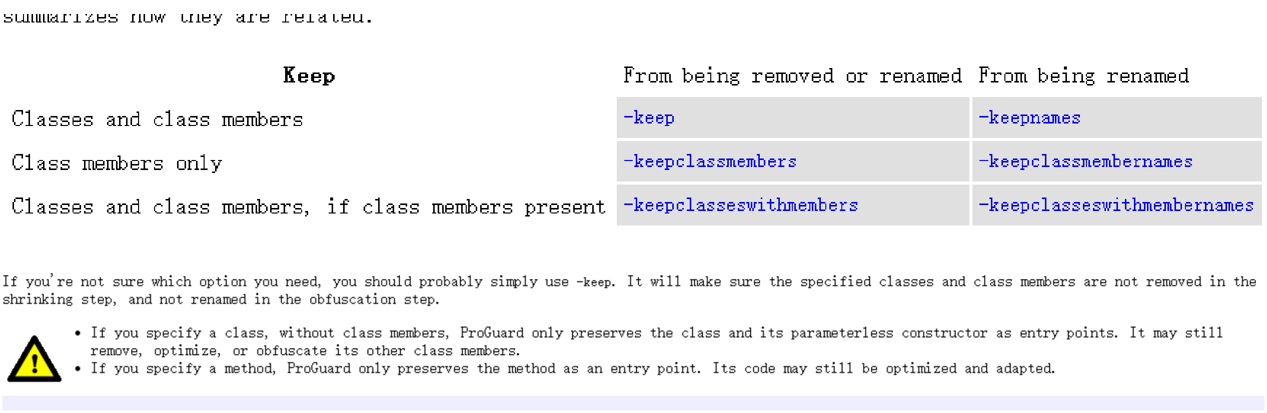
-dontwarn [class_filter] 不提示 warnning

-ignorewarnings 忽略警告

移除 log

```
-assumenosideeffects class android.util.Log {
   public static boolean isLoggable(java.lang.String, int);
   public static int v(...);
   public static int i(...);
   public static int w(...);
   public static int d(...);
   public static int e(...);
}
```

Keep and KeepName

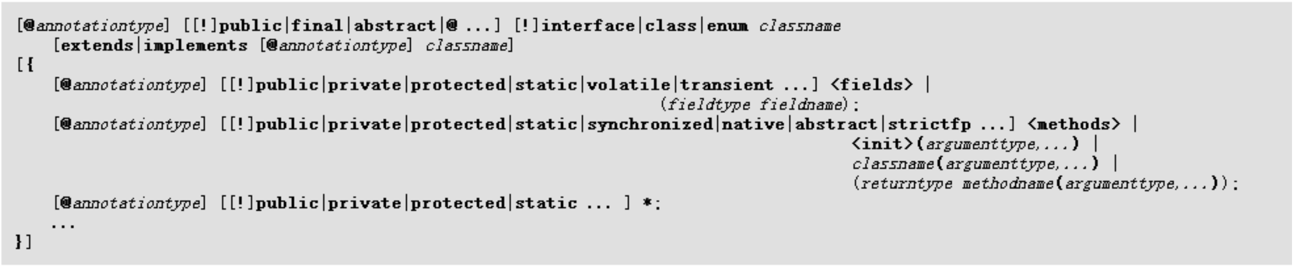


summarizes how they are related.

Keep	From being removed or renamed	From being renamed
Classes and class members	-keep	-keepnames
Class members only	-keepclassmembers	-keepclassmembernames
Classes and class members, if class members present	-keepclasseswithmembers	-keepclasseswithmembernames

If you're not sure which option you need, you should probably simply use -keep. It will make sure the specified classes and class members are not removed in the shrinking step, and not renamed in the obfuscation step.

⚠
- If you specify a class, without class members, ProGuard only preserves the class and its parameterless constructor as entry points. It may still remove, optimize, or obfuscate its other class members.
- If you specify a method, ProGuard only preserves the method as an entry point. Its code may still be optimized and adapted.



```
[@annotationtype] [[!]public|final|abstract|@ ...] [!]interface|class|enum classname
    [extends|implements [@annotationtype] classname]
[{
    [@annotationtype] [[!]public|private|protected|static|volatile|transient ...] <fields> |
                                                                      (fieldtype fieldname);
    [@annotationtype] [[!]public|private|protected|static|synchronized|native|abstract|strictfp ...] <methods> |
                                                                                  <init>(argumenttype,...) |
                                                                                  classname(argumenttype,...) |
                                                                                  (returntype methodname(argumenttype,...));
    [@annotationtype] [[!]public|private|protected|static ... ] *;
    ...
}]
```

Eclipse 默认的 proguard-android.txt（双击右侧打开）　proguard-android.txt

注意事项

混淆注意，以下类不能混淆

(1)避免混淆泛型 -keepattributes Signature

(2)排除反射、序列化相关的类

(3)JNI 中调用的类

(4)AndroidManifest.xml 中配置的类

其他需要注意

1.class 与 interface 以及 enum，class 包含 interface 与 class，另外两个关键字则严格限制类型

2.extends 与 implements 不包含父类

调试

1.dump.txt　apk 包内所有 class 的内部结构

2.mapping.txt　　混淆前后的映射

3.seeds.txt 未混淆的类和成员

4.usage.txt 列出从 apk 中删除的代码

还原日志 retrace.bat|retrace.sh [-verbose] mapping.txt [<stacktrace_file>]

比如 retrace.bat -verbose mapping.txt obfuscated_trace.txt

字符匹配

？匹配任意单个字符，非包分割符 keep class i?

*匹配出包分隔符以及目录分隔符外的所有符号

**匹配多个字符（任意）

！不匹配

比如"foo,*bar"匹配 foo 以及任意以 bar 结果的字符串

方法以及函数

<init>匹配所有构造函数

<fields>匹配所有字段

<methods>匹配所有方法

*匹配任何方法与字段

函数参数

%匹配所有基本类型（不包含 void）

?匹配类的单个字符 只针对类

*以及**类似之前的 只针对类

***任何类型

...任何个数的任何类型的参数