Jeb plugin/script report of PoRE

Student ID　20307130386
Name　杨舒晗

- **Task**
- **Which part of Jeb do you plan to improve?**

编写 jeb script 加载在 deguard 中上传 apk 后获得的显示类和方法的可能真实姓名的 mapping.txt，相应地重命名 apk 中的这些类和方法，同时修改变化过的包名

- **How does your plugin work?**

1、脚本设计

类 DeguardRenameClass 与脚本文件同名，类中 run 函数为脚本的入口点，异步调用类 LoadMapping，在类 LoadMapping 的 run 函数中根据 mapping.txt 进行重命名。先打开 mapping.txt 获得文件数据，后利用 IDexUnit 的 getClasses()和 getPackages()分别获得项目中的所有类和包的列表 classes 和 pkgs。

在遍历 classes 的过程中进行重命名。由于文件中类名在行中顶格，而类中的变量和函数会在紧接着类名的下一行，并且缩进四个空格，可以利用这一点，只对开头没有空格的行判断类名是否命中，并且只遍历文件中包含这个类的局部的行判断类中的函数是否需要修改。

```
androidx.savedstate.c -> androidx.savedstate.d
    androidx.savedstate.a c() -> c
androidx.savedstate.a -> androidx.savedstate.Item
    i.b a -> i
    android.os.Bundle b -> a
    boolean c -> b
```

通过 getAddress()获得类的 address，并处理成文件中的格式。若类的 address 在文件中出现，则利用 rename_method 函数处理类中的函数重命名；若类的 address 在文件中出现且有修改，则分别判断包名是否修改和类名修改，若修改

相应调用 rename_pkg 函数修改包名, 调用 comment 和 rename 函数备注并修改类名。

遍历过程中会出现某个类的包名在访问到之前同一个包下的类时已经修改过，这时若类的包名能匹配修改后的包名且需要修改类名，则调用 comment 和 rename 函数备注并修改类名，并且调用 rename_method 重命名类中的函数。

1）rename_pkg 函数

由于 IDexUnit 的 getpackages()获得的包的列表中，每个包只有包名的信息，而没有各个包的层级信息。但是观察输出的列表发现，类似深度优先搜索，对于每个包，会先遍历输出子包（子文件夹）的包名，然后才会输出同一层级的下一个包的包名，因此，对于列表中相同的包名，下标距离上一层包名更近的包为正确的包，而最外层的包中不会有重复的包名，可以直接修改；而对于多层包名的情况，通过二维数组统计类的 address 的每一层包名在列表中的 index，逐层定位距离上层包名更近的包，最终获得需要修改的包，用 rename 函数重命名

2）rename_method 函数

当类名在映射文件中某行命中后, 遍历该行之后的行直到遇到下一个字符顶格的行为止，若类中的函数名命中，则调用 comment 和 rename 函数备注并修改函数名。

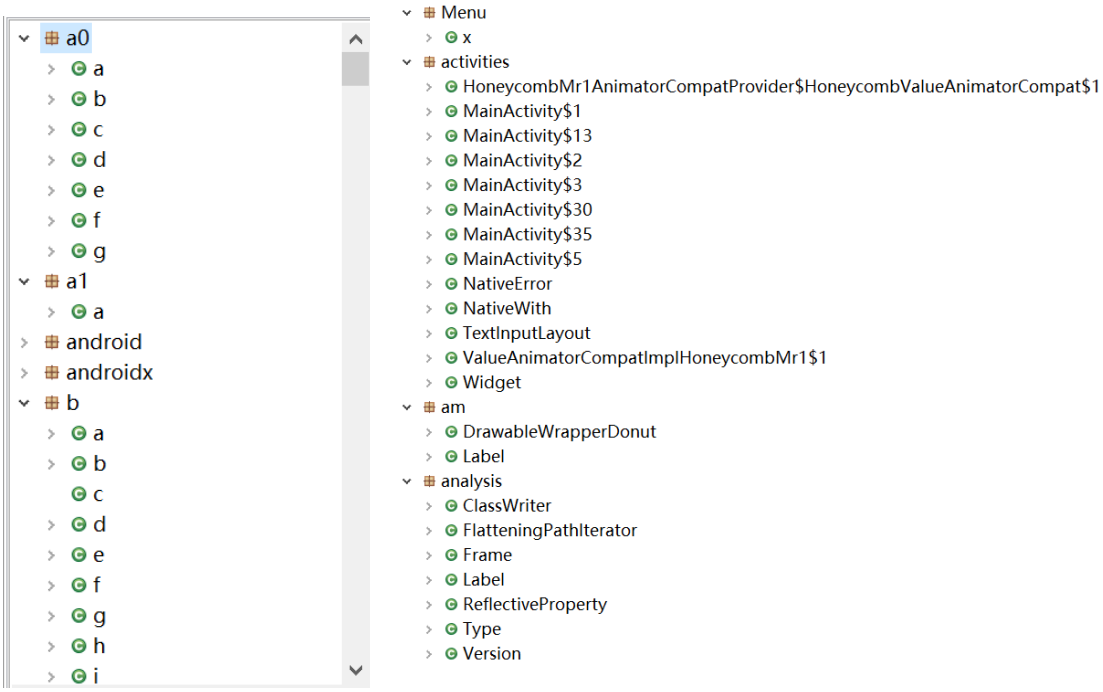
3）rename 函数和 comment 函数

调用 ActionContext 和 ActionRenameData 的构造函数，用 ActionRenameData 的 setNewName 设置新名称或用 setNewComment 设置新的注释，若 IDexUnit 的 prepareExecution 函数返回值为真，调用 executeAction 重命名或修改备注。

- **Screenshots of your plugin's execution result.**

使用 lab9 的 payload.apk 进行演示，效果如下：

1、修改包名类名



在日志打印修改的信息

```
rename to firstrun success!
comment to book.h success!
rename to SystemProperty success!
comment to android.view.View A(android.view.LayoutInflater,android.view.ViewGroup,android.os.Bundle) success!
rename to onCreateView success!
comment to void C(android.os.Bundle) success!
rename to add success!
comment to androidx.recyclerview.widget.LinearLayoutManager O() success!
rename to values success!
comment to void P(int) success!
rename to b success!
comment to void Q(com.google.android.material.datepicker.o) success!
rename to a success!
comment to void R(int) success!
rename to a success!
comment to void z(android.os.Bundle) success!
rename to onCreate success!
comment to boolean O(android.content.Context) success!
rename to a success!
comment to boolean P(android.content.Context,int) success!
rename to a success!
comment to com.google.android.material.datepicker.n success!
rename to Widget success!
comment to asm.u success!
rename to Color3f success!
comment to asm.v success!
rename to x success!
Done
```

2、修改函数名

mapping.txt 中新旧函数名映射关系如下：

```
com.google.android.material.transformation.ExpandableBehavior -> com.google.android.material.transformation.ExpandableBehavior
    int a -> mState
    boolean d(androidx.coordinatorlayout.widget.CoordinatorLayout,android.view.View,android.view.View) -> onDependentViewChanged
    boolean h(androidx.coordinatorlayout.widget.CoordinatorLayout,android.view.View,int) -> a
    boolean s(boolean) -> get
    boolean t(android.view.View,android.view.View,boolean,boolean) -> update
```

修改效果如下:

```
@Override  // androidx.coordinatorlayout.widget.CoordinatorLayout$c
public boolean onDependentViewChanged(CoordinatorLayout arg2, View arg3, View arg4) {
```

```
@Override  // androidx.coordinatorlayout.widget.CoordinatorLayout$c
public boolean a(CoordinatorLayout arg6, View arg7, int arg8) {
```

```
// boolean s(boolean)
public final boolean get(boolean arg4) {
```

```
// boolean t(android.view.View,android.view.View,boolean,boolean)
public abstract boolean update(View arg1, View arg2, boolean arg3, boolean arg4);
```