在构建过程中自动打皮肤包

感觉价值不大。原因如下：

1. 如果在**feature**流水线上使用，即使自动打包皮肤包以后，还需要手动修改 DarkModeUtils 中的皮肤包版本号。虽然可以想办法自打皮肤包的时候修改 DarkModeUtils.class 合并到dex之前修改 DarkModeUtils.class，但是感觉性价比太低。
2. 打了皮肤包以后，需要将皮肤包文件推送到代码仓库。
3. 主要的原因：现在项目中的 make\_darkmode.sh 用起来已经比较方便了。

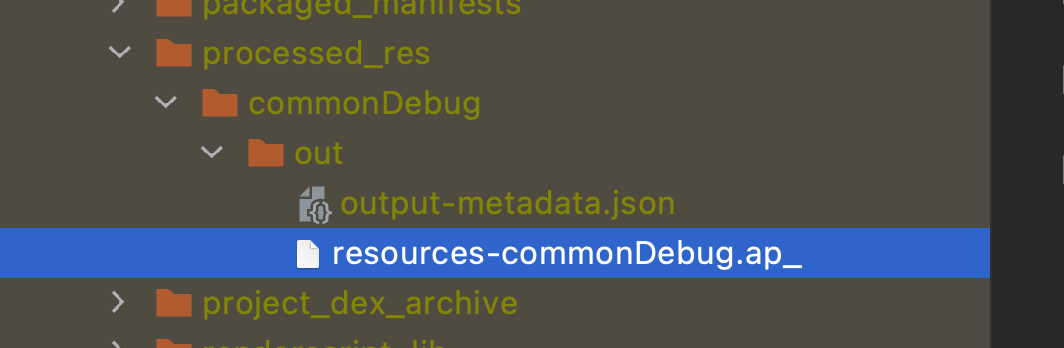
所以：本文档只作为一个记录，记录一下是如何实现的自动打皮肤包的。代码分支是 [代码分支](https://git.woa.com/YuewenGroup/XXReader/xxreader_client/xxreader_client_android/XXReader_Android/tree/feature_auto_package_darkmode) 。

# 思路：

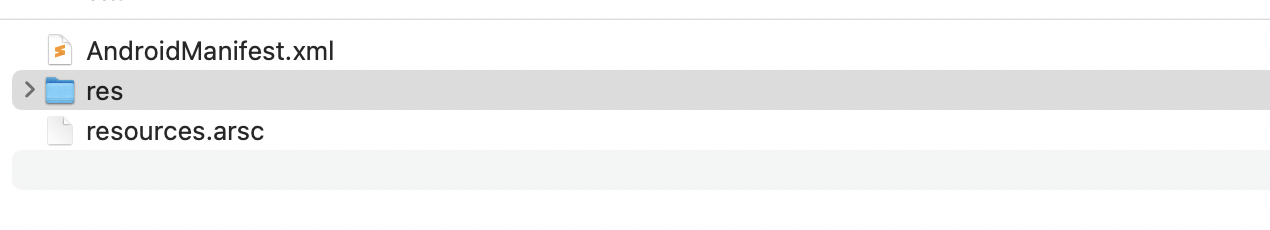
自定义一个插件，在编译完所有的resouces资源以后和合并assets资源之前，插入一个ReplaceTask。

该Task用来把编译后的资源中的**color-night-v8**文件夹下的颜色替换老的**darkmode.7z**中的color文件夹下的所有颜色，然后生成新的darmode.7z文件。

在编译完所有的resouces资源以后，会生成一个压缩文件。**resources-commonDebug.ap\_**



把这个压缩文件解压一下

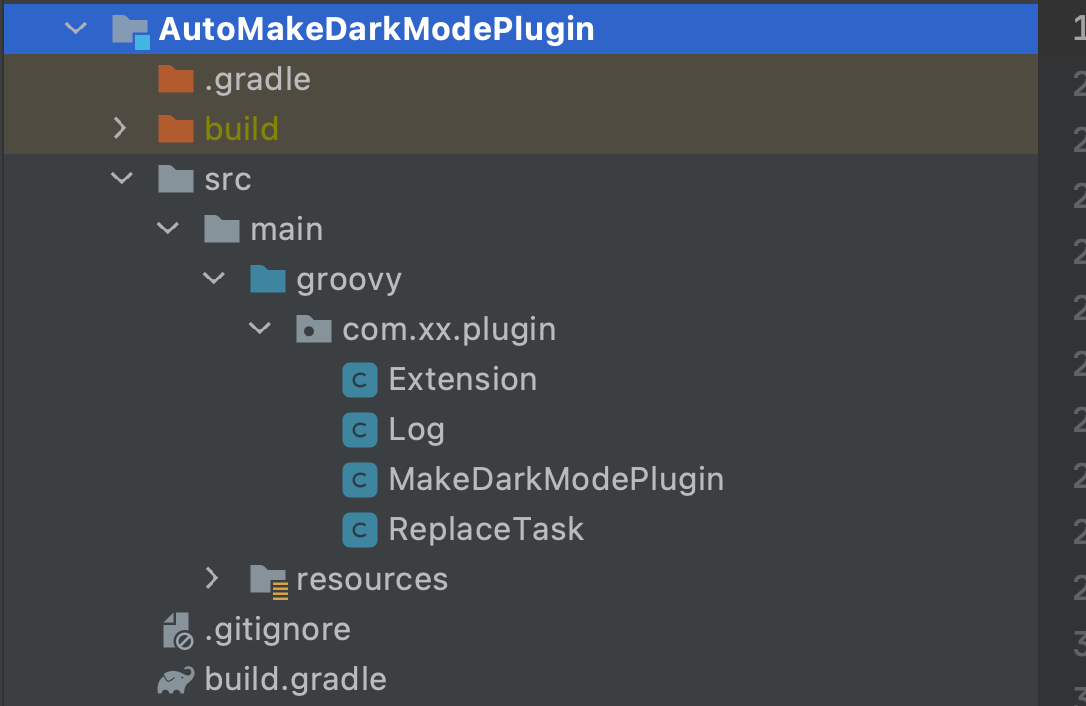


res/color-night-v8下面的文件就是所有的夜间颜色资源。



下面开始实现插件的步骤

首先在项目中创建一个module，作为插件模块。



先定义一个Extension类，让用户可以选择是否开启自动打皮肤包的功能。

插件类 MakeDarkModePlugin

* 注释0处，如果关闭了自动打皮肤包，直接return。
* 注释1处，debug打包的时候，命令是 assembleCommonDebug。
* 注释2处，processCommonDebugResources
* 注释3处，mergeCommonDebugAssets
* 注释4处，让 replaceTask 依赖 processResourcesTask，也就是说 replaceTask 会在 processResourcesTask执行完毕之后执行。保证replaceTask任务执行的时候，所有的资源都编译完毕了。
* 注释5处，让 mergeAssetsTasks 依赖 processResourcesTask, replaceTask，保证在merge assets之前，我们已经替换完了 assets目录下的内容。

接下来看一下 ReplaceTask 。

@TaskAction 注解的方法在task执行的时候会被调用。

* 注释1处 将 resources-commonDebug.ap\_ 解压。
* 注释2处，把老的皮肤包解压到 newSkinDirPath 目录 assets/newSkin 下
* 注释3处， 把compliedRes/res/color-night-v8下面的文件都拷贝到 assets/newSkin/color 目录下，直接替换老的颜色值。
* 注释4处，把 newSkin 目录打包成 darkmodeNew.7z
* 注释5处，把 darkmode.7z 重命名为 darkModeBack.7z
* 注释6处，把 darkmodeNew.7z 重命名为 darkmode.7z
* 注释7处，删除临时文件夹