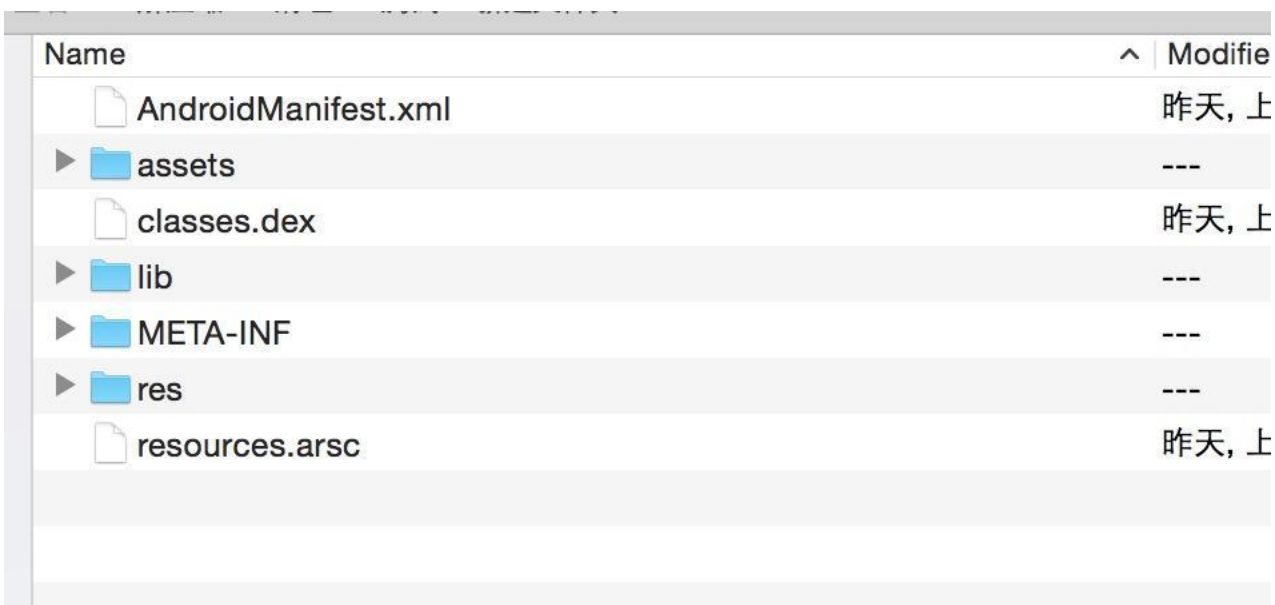
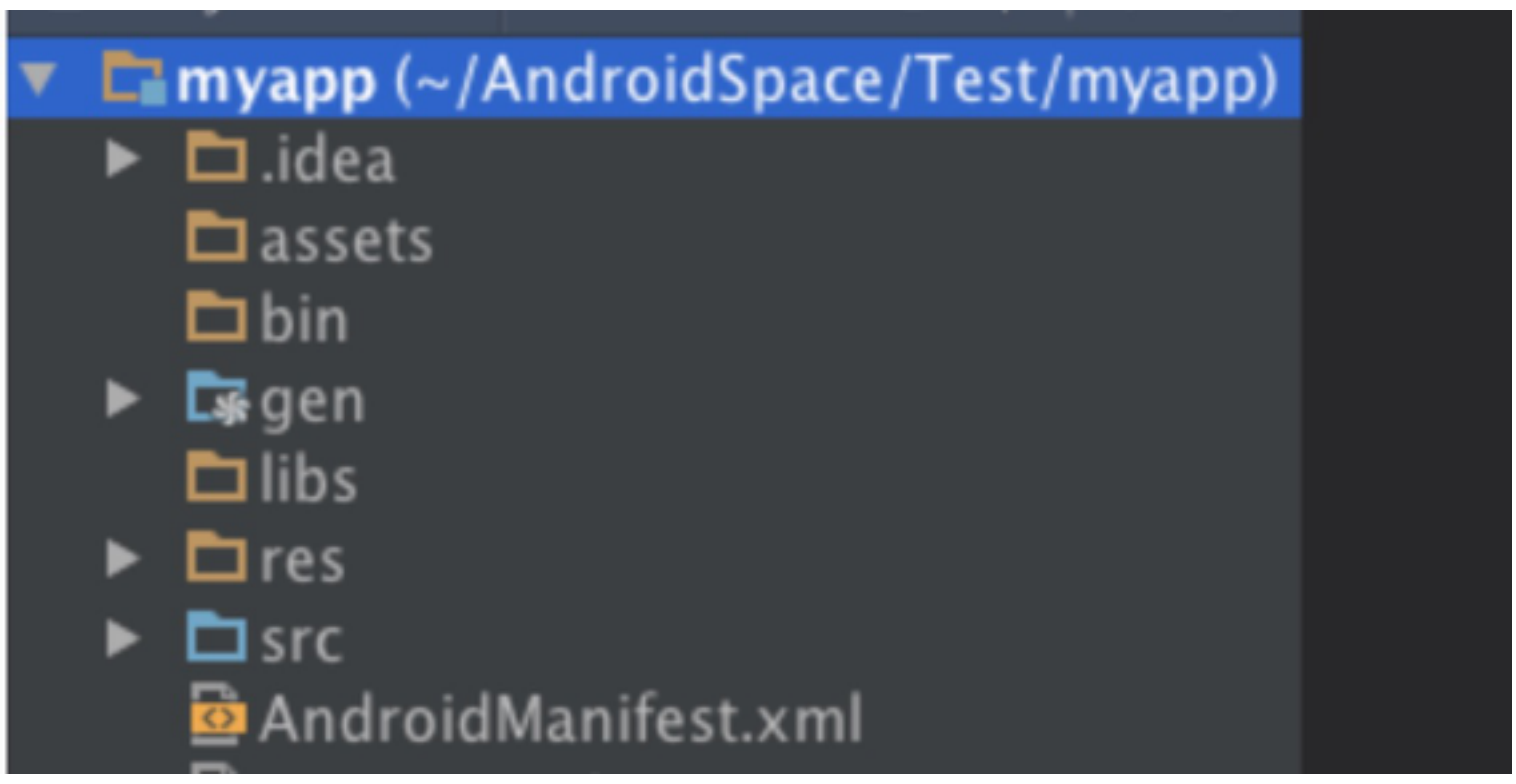
Android的安装文件为apk，要逆向Android应用需要先了解apk文件的结构。使用rar或好压之类的压缩软件解压之后可以发现apk的文件结构大致包含如下内容：



包含三个文件：AndroidManifest.xml，classes.dex，resources.arsc和四个文件夹assets，lib，res，META-INF；
如果有一定的Android开发经验，我们知道一个Android项目中一般包含如下结构：

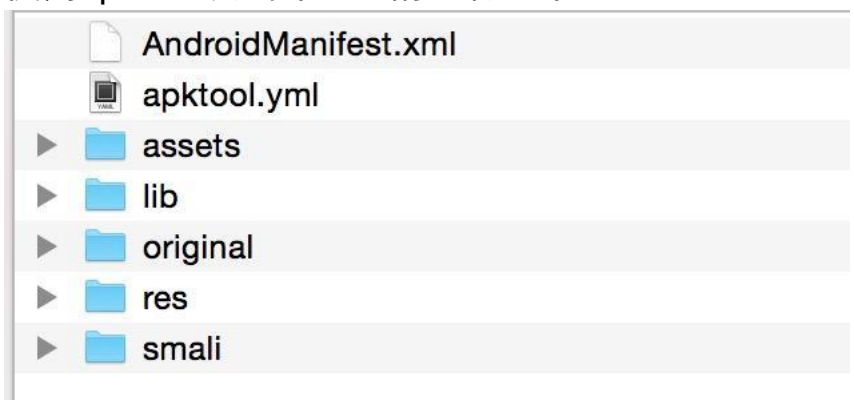


其中对应关系如下：

	apk结构	项目结构
1	AndroidManifest.xml	AndroidManifest.xml
2	lib	libs
3	res	res
4	assets	assets
5	resources.arsc	res
6	classes.dex	src/gen/libs

AndroidManifest.xml是不能直接使用文本工具打开的，这里的AndroidManifest经过了编译过程，更加节省控件也更方便操作系统执行；
lib目录对应于项目结构中的libs下的so文件，so为c/c++写的在linux下的可执行文件（elf格式）；
res文件包含项目res文件下的drawable，layout和图片文件，其中drawable和layout也是经过编译的
assets目录下跟原来的assets目录下文件是相同的
resources.arsc文件主要是res中的字符串string.xml的编译后的结果。
classes.dex是所有java代码的编译结果，包括自己项目中的.java文件，自动生成的gen文件夹下的java文件，引入的jar文件
META-INF为签名文件，在项目打包apk的时候生成，里面的内容是对每个文件的签名信息，应用安装时会校验每个文件的签名信息，如果校验信息不正确是不能安装成功的。同时，如果同一个apk不同的签名信息进行签名，是不能覆盖安装的。

使用apktool反编译之后的结构如下



其与apk包结构的对应关系如下：

	apk结构	apktool反编译
1	AndroidManifest.xml	AndroidManifest.xml
2	lib	lib
3	res	res
4	assets	assets
5	resources.arsc	res
6	classes.dex	smali

经过apktool反编译之后，可以发生了如下的变化：
1、所有的xml都可以使用文本编辑器查看了，包括AndroidManifest.xml，res下的drawable和layout文件；
2、res中增加了string.xml，public.xml等一些xml文件；
3、增加了一些smali文件；
通过以上的对应表我们可以知道resources.arsc中的内容为反编译后的string.xml和public.xml文件，classes.dex为所有的smali代码。
4、反编译后多了一个original文件夹，该文件夹下包含AndroidManifest.xml和META-INF文件，这个是apk结构中原来的AndroidManifest.xml河META-INF的备份；

如上介绍了apk的结构、包括项目结构和apktool反编译之后的结构；从上述描述中可以理解如下两点：
1、assets目录下的资源文件不会被编译；res的资源文件都会被编译的；可知如果想要原来的apk中添加一些自己的文件而不被编译影响的话，可以添加到assets目录中，比如各种破解网站自己的启动页所需要的资源；
2、直接使用一个文件压缩工具修改一个apk之后是不能直接运行的，原因在于签名，apk的任何修改都会改变该文件的信息，对应的签名验证就不能通过；因此修改apk之后后在对其进行签名就可以正常安装了。