

# Android 单机小应用程序开发实验

实验名称|Android 应用程序打包

实验日期|2017-04-22

# 一 实验内容

将一个 Android 应用程序打包为 apk 安装包的过程写成相应的文档（截图+说明），能使有基本计算机操作基础的人看着文档就能顺利完成相应操作。

## 二 实验环境

操作系统：

Windows 7 64-bit

开发环境：

Android Studio 2.2.3

运行环境：

设备型号：Coolpad 8297D

Android 版本：4.2.2

内核版本：3.4.39

## 三 实验过程

### 什么是 apk 打包？

首先，apk 本身就是一种压缩包，即 Android application package。我们编写的 Java 代码，用到的资源，以及相关的配置文件最终都被打包进一个 apk 中。我们可以将一个 apk 后缀名改为.zip 并打开观察一下它的内容：

res			文件夹
META-INF			文件夹
resources.arsc	1,588	1,588	ARSC 文件
classes.dex	1,824	897	DEX 文件
AndroidManifest.xml	1,468	568	XML 文件

其中 res 目录存放着布局信息以及各种其他资源，如图标等；META-INF 包含一些元信息（meta-information）：

MANIFEST.MF	701	383	MF 文件
FIRSTONE.SF	822	446	SF 文件
FIRSTONE.RSA	1,338	1,074	RSA 文件

MANIFEST.MF 是 Java 代码进行 Jar 打包时生成的；后面两个文件是和 apk 签名有关的文件。

resources.arsc 是一个资源索引表，做资源 ID 和资源之间的对应。

classes.dex 是 Java 字节码(.class)经过交叉编译生成的 dalvik 虚拟机字节码。

AndroidManifest.xml 则包含了整个应用的元数据。

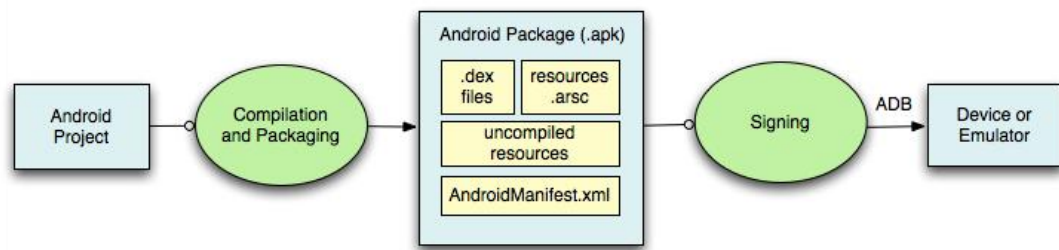
关于 apk 签名，这里不再展开叙述，可以参考【参考资料】中第五、八链接。

我们这里仅仅对 apk 内容做概括性的介绍，既不深入其中机制，也不打算讲述为什么是这样布置，以及一个 apk 怎样在设备上运行。

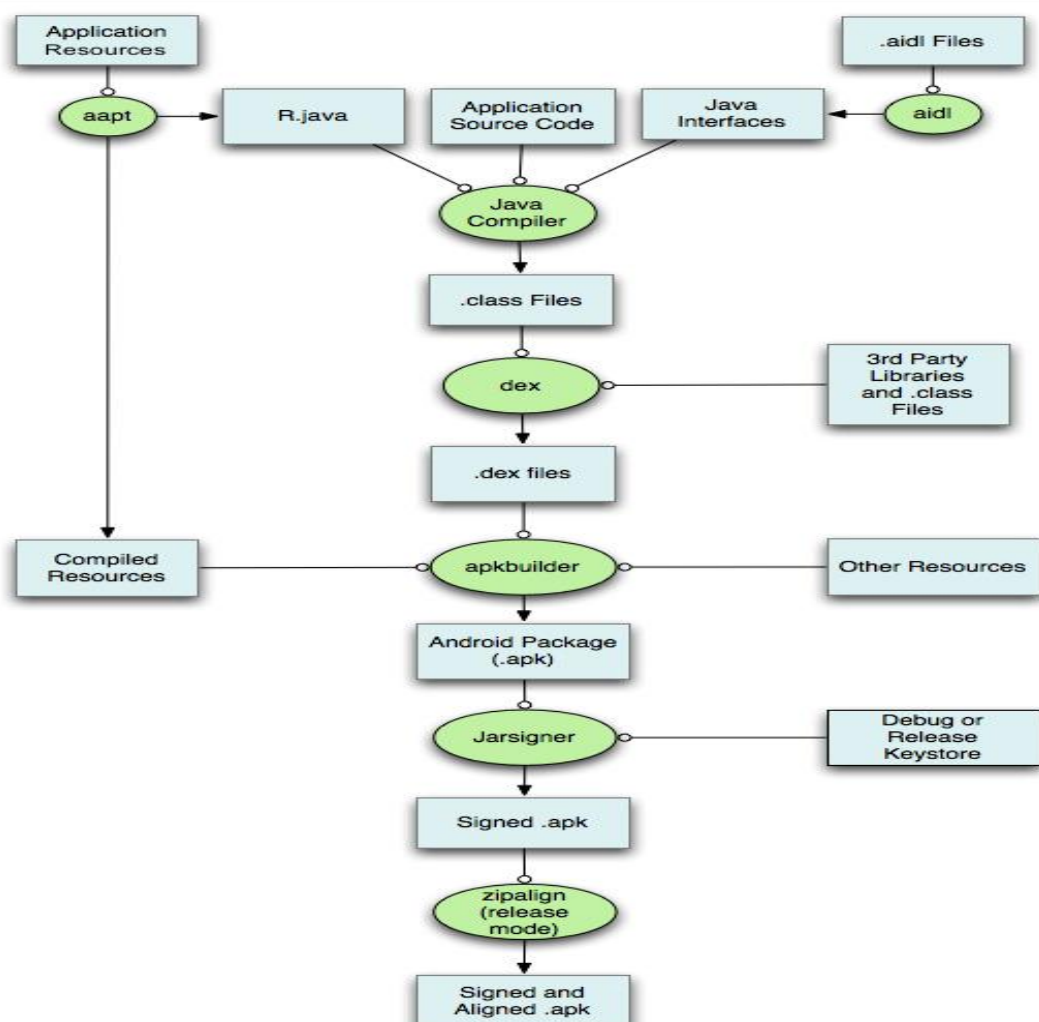
接下来介绍打包的流程，思路是先放总体的流程图片，再给

出文字说明。在下一部分，我们将具体给出两种打包方法并附实践操作。

先看一个概括的流程：



再看一个充满细节的流程：



上图中的椭圆形代表工具，方形代表相应文件。需要注意的是，目前新版本的 SDK 已经不再使用 apkbuilder 脚本生成 apk，

而是将此功能并入了 aapt 中。总结一下，apk 打包有下面几个步骤：

- 根据资源的使用生成 R.java
- 把所有.java 代码编译为.class 文件
- 把.class 文件打包成.jar 文件
- 把所有.jar 文件交叉编译成.dex 文件
- 把所有资源文件打包成 apk 包
- 把 classes.dex 添加到上步 apk 包
- 对 apk 进行签名
- 进行对齐优化

## 如何进行打包？

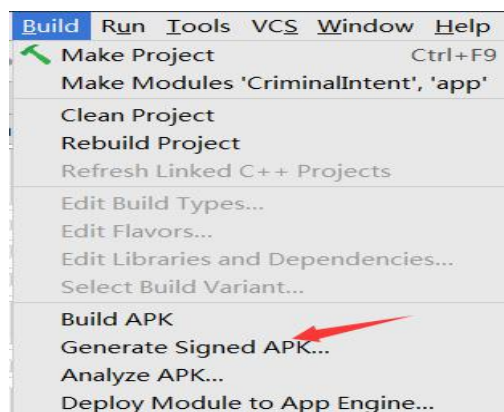
### 【使用 Android Studio 进行 apk 打包】

其实我们在 Android Studio 中点击 Run ‘app’ 后进行真机调试时已经生成了一个 apk，并自动安装到了你的设备上。我们也可以在项目目录下找到它，具体到实验环境中，就是下面这个目录：

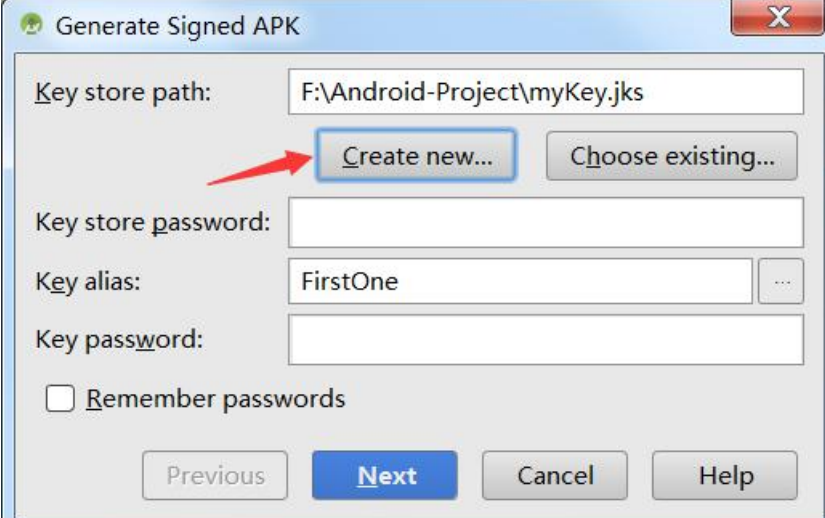
计算机 ▶ disc F (F:) ▶ Android-Project ▶ CriminalIntent ▶ app ▶ build ▶ outputs ▶ apk

可以在该目录下找到 app-debug.apk。这个 apk 使用了 debug key 签名。我们在分发 apk 给用户时，应用必须以 release key 签名。下面将讲述这个过程：

①



②



The "Generate Signed APK" dialog box is shown. It has a title bar with a green icon and a close button. The main area contains several fields and buttons. A red arrow points to the "Create new..." button.

Key store path: F:\Android-Project\myKey.jks

Key store password: [empty field]

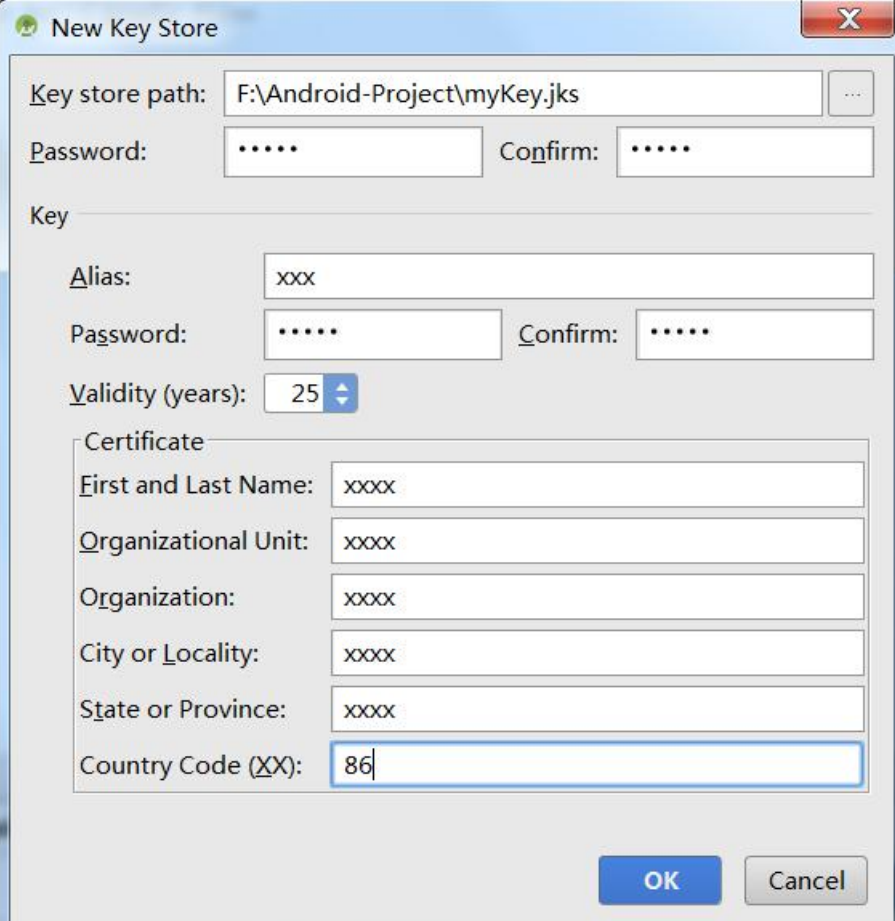
Key alias: FirstOne

Key password: [empty field]

☐ Remember passwords

Buttons: Previous, Next, Cancel, Help

③



The "New Key Store" dialog box is shown. It has a title bar with a green icon and a close button. The main area contains several fields and buttons. The "Key" section is expanded.

Key store path: F:\Android-Project\myKey.jks

Password: [empty field] Confirm: [empty field]

Key

Alias: xxx

Password: [empty field] Confirm: [empty field]

Validity (years): 25

Certificate

First and Last Name: xxxx

Organizational Unit: xxxx

Organization: xxxx

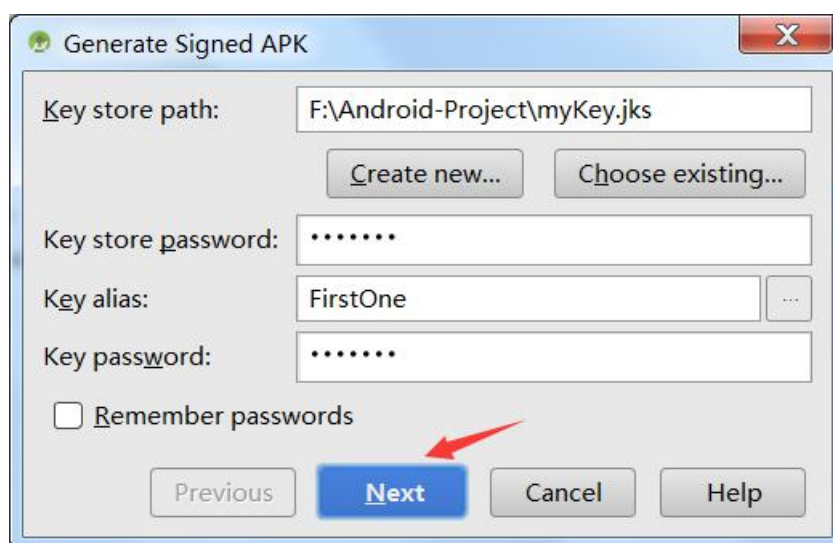
City or Locality: xxxx

State or Province: xxxx

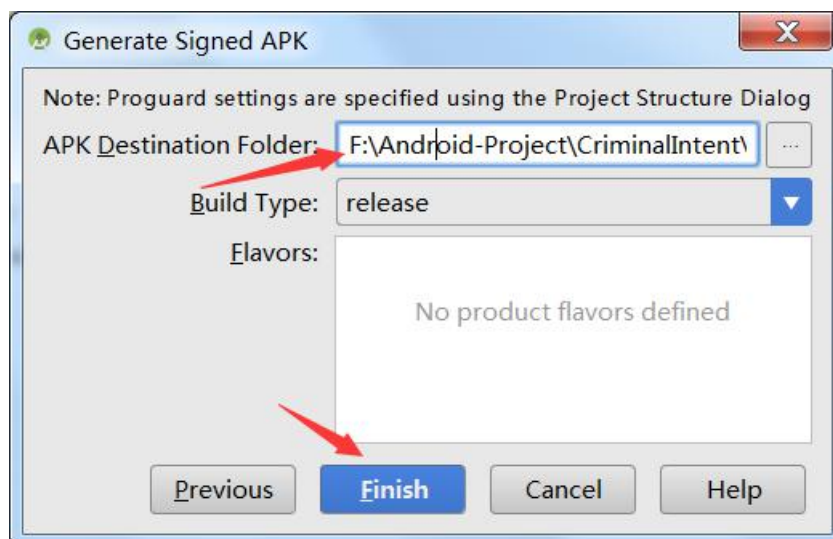
Country Code (XX): 86

Buttons: OK, Cancel

④



⑤



好了。等待编译、打包成功后，可以在你设定的目标文件夹中找到 app-release.apk。

### 【使用命令行进行 apk 打包】

IDE 封装了一切，然而我们偏要敲开冰面，看看水下是什么——原来别有洞天。后面内容主要学习自【参考资料】第四个（好文章，强烈推荐阅读），我们将通过从零构建一个 Android 工程并最终在设备上运行它来讲解所有的知识点。首先 show 一下最



后运行的结果，它是经典的 helloworld 程序（也就是在建立工程后没有添加任何其他代码的默认程序）：



Are you ready? Let's go :)

注：

在命令行中进行操作总是要涉及到环境变量的问题，因为我们希望能够直接输入工具名称就可以直接使用，而不必每次都输入它的完整路径。为此，我们首先在 **Path** 环境变量中添加以下路径（关于添加环境变量的方法，请自行搜索）：

%ANDROID\_HOME%\platform-tools\

%ANDROID\_HOME%\build-tools\25.0.1\

%ANDROID\_HOME%\tools\

%JAVA\_HOME%\bin\

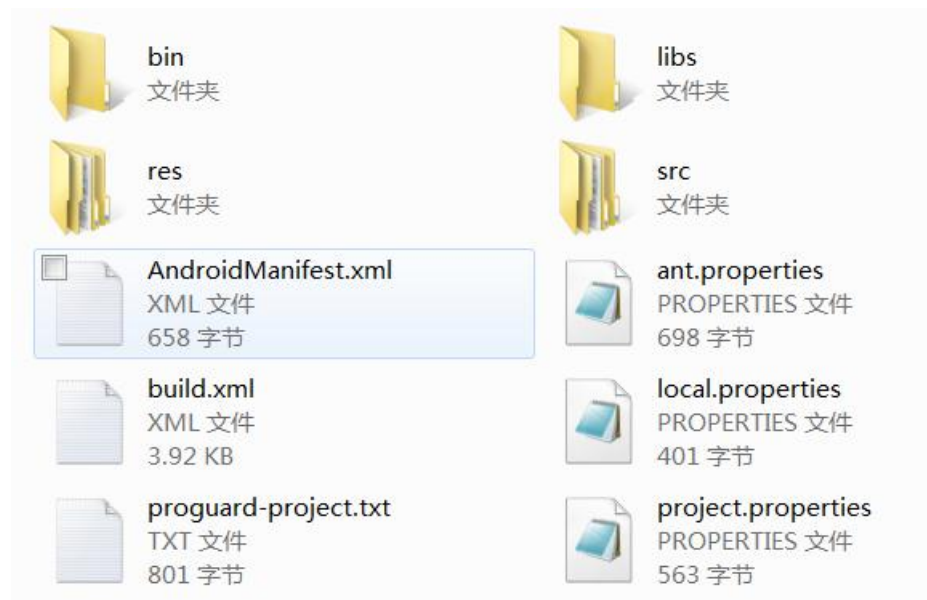
其中%ANDROID\_HOME%为 Android SDK 的路径,%JAVA\_HOME%为 JDE 的路径。另外, build-tools 下请以自己开发所用的实际版本为准, 我们这里是 25.0.1。

另外, 后面的操作需要用到一个小技巧, 就是在当前窗口展示的文件夹内打开命令行窗口: 在空白处按住 **shift** 并点击右键选择“在此处打开命令窗口”。

我们新建一个空白目录 **test**, 并在该目录下打开命令行窗口, 输入以下命令, 新建一个标准 Android 工程:

```
android create project --activity MainActivity --package  
com.example.www --path ./AndroidTestProject -t android-21
```

此时会生成一个 **AndroidTestProject** 目录, 其中有:



我们 **cd** 进入这个目录, 并在其中新建一个 **gen** 文件夹。

接着, 生成 **R.java**:

```
aapt package -m -J ./gen -M ./AndroidManifest.xml -S ./res  
-I %ANDROID_HOME%/platforms/android-21/android.jar
```

接着，编译.java 代码为.class 字节码：

```
javac -target 1.7 -source 1.7 -d ./bin/  
-bootclasspath %ANDROID_HOME%/platforms/android-21/android.jar ./src/com/example/www/MainActivity.java ./gen/com/example/  
www/R.java
```

需要注意的是，要把 R.java 也添加到命令中。接着，将.class 打包为.jar：

```
jar cvf ./bin/classes.jar -C ./bin .
```

接着，把.jar 交叉编译为.dex：

```
dx --dex --output ./bin/classes.dex ./bin/classes.jar
```

接着，新建 out 文件夹，然后将资源打包成.apk：

```
aapt package -f -M AndroidManifest.xml -S ./res  
-I %ANDROID_HOME%/platforms/android-21/android.jar  
-F ./out/res.apk
```

接着，cd 进入 bin 目录（这是因为 classes.dex 不能带路径），再将 classes.dex 添加进 apk：

```
aapt add ../out/res.apk classes.dex
```

接下来要对 apk 签名。在签名之前，我们先生成一个 keystore 文件，首先 cd .. 进入 AndroidTestProject 目录（你也可以放在其他目录），然后：

```
keytool -genkey -alias android.keystore -keyalg RSA  
-validity 20000 -keystore android.keystore
```

接着，使用刚刚生成的 keystore 对 apk 签名：

```
jarsigner -digestalg SHA1 -sigalg MD5withRSA -verbose  
-keystore ./android.keystore -storepass root123 -keypass  
root123 -signedjar ./out/reg.signed.apk ./out/res.apk  
android.keystore
```

最后，对 apk 包进行对齐优化：

```
zipalign -f 4 ./out/reg.signed.apk ./out/myfinal.apk
```

我们使用 adb 来进行 apk 的安装，这样就不需要自己手动在设备上点击安装了：

- ① 将设备连接到电脑
- ② 在我们的 apk 所在目录打开命令窗口
- ③ 输入 `adb install ./myfinal.apk`，稍等，出现以下信息说明安装成功：

```
[100%] /data/local/tmp/myfinal.apk  
      pkg: /data/local/tmp/myfinal.apk  
Success
```

## 四 实验总结

在实验开始之前，我们感觉并没有什么可写的，因为在 Android Studio 里点一个按钮就直接进行真机调试了，多点几下就能生成使用 release key 签名的 apk。就像大一时学 C++ 一样，在 Code::Blocks 里边按 F9 就可以进行编译链接运行。有句话说，人只能看到他想看到的东西。我们希望看到真相，所以便一层层抽丝剥茧。这也像后来到 Linux 下编写 C 语言程序，才知道 IDE 也是通过命令行来生成可执行文件的，而整个过程又可以分为“预编译”、“编译”、“汇编”、“链接”以及运行时的动态链接等。总之，保持初学者的心态，探索眼前的一切。

## 五 参考资料

- Android 逆向分析(2) APK 的打包与安装

<http://blog.zhaiyifan.cn/2016/02/13/android-reverse-2/>

- AndroidStudio 打包步骤

<http://www.jianshu.com/p/e69d0d0bd528>

- **Android** 编程权威指南（第 2 版）
- 也说 Android Apk 打包（好文）

<http://leenjewel.github.io/blog/2015/12/02/ye-shuo-android-apk-da-bao/>

- 怎么使用命令对 APK 包进行签名

<http://jingyan.baidu.com/article/3c48dd3491d91fe10be358f4.html>

- Android 应用程序安装过程解析(源码解析)

<http://blog.csdn.net/lpjishu/article/details/59527662>

- android 中 APK 包的安装以及 adb 命令的使用

<http://kanwoerzi.iteye.com/blog/1304251>

- 应用运营知识：应用的签名

<http://dev.xiaomi.com/doc/p=70/index.html>

- 如何生成 android 的 keystore 文件

<http://jingyan.baidu.com/article/59703552e877f98fc00740f0.html>