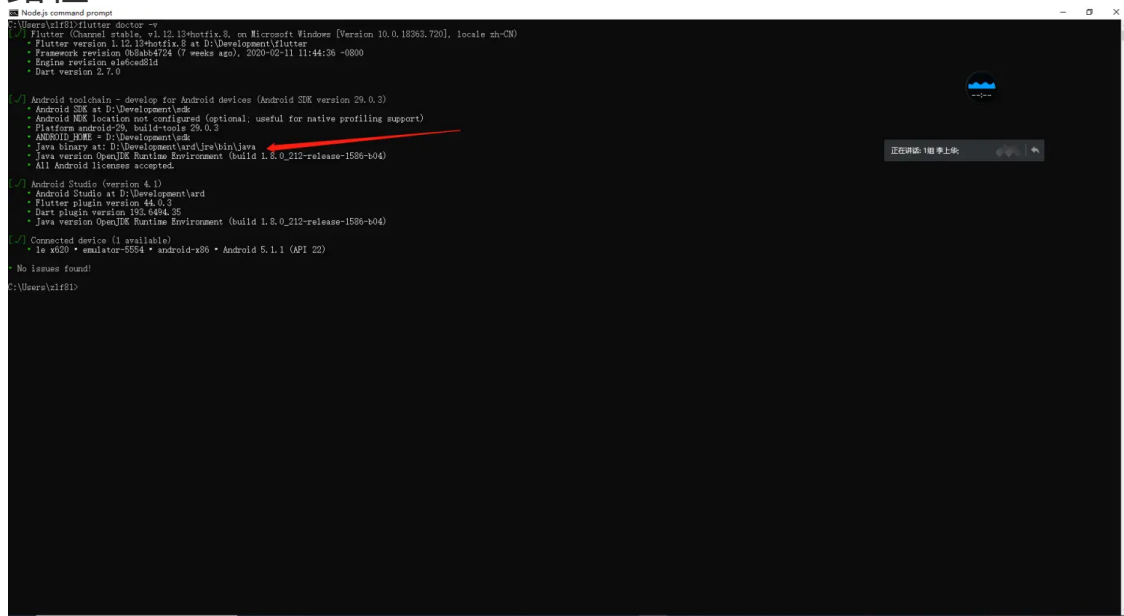


# Flutter 打包发布 android 项目配置

## 一、首次打包需要创建 Key

1.1 在命令行输入：flutter doctor -v 找到 Java binary 存放的路径



```
C:\Users\zif81>flutter doctor -v
Flutter (Channel stable, v1.12.1-hotfix.2, on Microsoft Windows [Version 10.0.18363.720], locale zh-CN)
  • Flutter version 1.12.1-hotfix.2 at D:\Development\Flutter
  • Framework revision 9b864724 (7 weeks ago), 2020-02-11 11:44:36 -0800
  • Engine revision a1dc631d
  • Dart version 2.7.0

✓ Android toolchain - develop for Android devices (Android SDK version 29.0.3)
  • Android SDK at D:\Development\sdk
  • Android NDK location not configured (optional, useful for native profiling support)
  • Platform android-29, build-tools 29.0.3
  • ANDROID_HOME = D:\Development\sdk
  • Java binary at: D:\Development\ard\jre\bin
  • Java version OpenJDK Runtime Environment (build 1.8.0_212-release-1586-b04)
  • All Android licenses accepted.

✓ Android Studio (version 4.1)
  • Android Studio at D:\Development\ard
  • Flutter plugin version 44.0.3
  • Dart plugin version 193.6484.35
  • Java version OpenJDK Runtime Environment (build 1.8.0_212-release-1586-b04)

✓ Connected device (1 available)
  • In x86 • emulator-5554 • android-x86 • Android 5.1.1 (API 22)

• No issues found!
C:\Users\zif81>
```

1.2 在当前 D:\Development\ard\jre\bin 文件夹下使用 cmd 命令行工具输入

```
keytool -genkey -v -keystore D:/key.jks -keyalg RSA
-keysize 2048 -validity 10000 -alias key
```

//-keystore 表示生成的签名文件的名称，后期发布项目的是会用到该文件

//-alias 后续项目发布的时候也会用到，可以根据自己的需要进行定制库别名

1.3 当运行命令的时候, 需要输入相关口令的密码和一些奇奇怪怪的消息比如个人信息直接回车就可以了, 然后中间会问你是否确认, 输入 y 就可以了不出意外在 D 盘的根目录下就会创建一个 key.jks 文件了, 请一定保存好密码, 后期会用到

\*\*\*\*\*注意: 保持 key.jks 文件的私密性, 不要将其加入到公共代码控制中, 注意在 .gitignore 中添加忽略文件

如图

```
D:\Development\ard\jre\bin>keytool -genkey -v -keystore D:/key.jks -keyalg RSA -keysize 2048 -validity 10000 -alias key
输入密钥库口令:
再次输入新口令:
您的名字与姓氏是什么?
[Unknown]: zhong
您的组织单位名称是什么?
[Unknown]:
您的组织名称是什么?
[Unknown]:
您所在的城市或区域名称是什么?
[Unknown]:
您所在的省/市/自治区名称是什么?
[Unknown]:
该单位的双字母国家/地区代码是什么?
[Unknown]:
CN=zhong, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown是否正确?
[否]: y
正在为以下对象生成 2,048 位RSA密钥对和自签名证书 (SHA256withRSA) (有效期为 10,000 天):
CN=zhong, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown
输入 <key> 的密钥口令
(如果和密钥库口令相同, 按回车):
[正在存储D:/key.jks]

Warning:
JKS 密钥库使用专用格式。建议使用 "keytool -importkeystore -srckeystore D:/key.jks -destkeystore D:/key.jks -deststoretype pkcs12" 迁移到行业标准格式 PKCS12。
D:\Development\ard\jre\bin>
```

## 二、进行一些相关的配置

2.1 在你开发的项目下的 android 目录下创建一个 key.properties 文件 输入以下代码,保存一下

```
storePassword= 刚刚创建密钥库时的密码
keyPassword= 刚刚创建密钥的密码
keyAlias=key //库的别名
storeFile=D:/key.jks //key.jks文件路径
```

2.2 打开 flutter 的 /android/app/build.gradle 文件, 在 android 前加入

```
def keystoreProperties = new Properties()
```

```

def keystorePropertiesFile =
rootProject.file('key.properties')
if (keystorePropertiesFile.exists()) {
    keystoreProperties.load(new
FileInputStream(keystorePropertiesFile))
}

```

```

运行(R) 终端(T) 帮助(H) build.gradle - Flutter - Visual Studio Code
shopPage.dart classPage.dart detail.dart ! pubspec.yaml main.dart cart.dart AndroidManifest.xml build.gradle X
hello_flutter > android > app > build.gradle
1
2  def keystoreProperties = new Properties()
3  def keystorePropertiesFile = rootProject.file('key.properties')
4  if (keystorePropertiesFile.exists()) {
5      keystoreProperties.load(new FileInputStream(keystorePropertiesFile))
6  }
7
8  def localProperties = new Properties()
9  def localPropertiesFile = rootProject.file('local.properties')
10 if (localPropertiesFile.exists()) {
11     localPropertiesFile.withReader('UTF-8') { reader ->
12         localProperties.load(reader)
13     }
14 }
15
16 def flutterRoot = localProperties.getProperty('flutter.sdk')
17 if (flutterRoot == null) {
18     throw new GradleException("Flutter SDK not found. Define location with flutter.sdk in the local.properties file.")
19 }
20
21 def flutterVersionCode = localProperties.getProperty('flutter.versionCode')
22 if (flutterVersionCode == null) {
23     flutterVersionCode = '1'
24 }
25
26 def flutterVersionName = localProperties.getProperty('flutter.versionName')
27 if (flutterVersionName == null) {
28     flutterVersionName = '1.0'
29 }
30
31 apply plugin: 'com.android.application'
32 apply plugin: 'kotlin-android'
33 apply from: "$flutterRoot/packages/flutter_tools/gradle/flutter.gradle"
34
35 android {
36     compileSdkVersion 28
37

```

把 buildTypes 这个对象替换成

```

signingConfigs {
    release {
        keyAlias "创建的密钥库别名，如果不知道可以把D:/key.js
文件粘贴到（上图1.1文件里面）打开命令行工具输入keytool -list
-v -keystore key.jks -storepass 密钥的密码查看库名"
        keyPassword "密钥的密码"
    }
}

```

```

        storeFile file("密钥所在文件夹，如果是按照我上面的步
骤的话是D:/key.jks")
        storePassword "密钥库的密码"
    }
}
buildTypes {
    release {
        signingConfig signingConfigs.release
    }
}

```

如图

```

    }
    signingConfigs {
        release {
            keyAlias="key"
            keyPassword="keytool"
            storeFile file('D:/key.jks')
            storePassword="keytool"

            // keyAlias keystoreProperties['keyAlias']
            // keyPassword keystoreProperties['keyPassword']
            // storeFile file(keystoreProperties['storeFile'])
            // storePassword keystoreProperties['storePassword']
        }
    }

    buildTypes {
        release {
            // TODO: Add your own signing config for the release build.
            // Signing with the debug keys for now, so `flutter run --release` works.
            signingConfig signingConfigs.release
        }
    }
    // buildTypes {
    //     release {
    //         // TODO: Add your own signing config for the release build.
    //         // Signing with the debug keys for now, so `flutter run --release` works.
    //         signingConfig signingConfigs.debug
    //     }
    // }
}

```

查看密钥库别名：

```

D:\Development\ard\jre\bin>keytool -list -v -keystore key.jks -storepass keytool
密钥库类型: jks
密钥库提供方: SUN

您的密钥库包含 1 个条目

别名: key
创建日期: 2020-4-3
条目类型: PrivateKeyEntry
证书链长度: 1
证书[1]:
所有者: CN=zhong, OU=Unknown, O=Unknown, L=changping, ST=beijing, C=cn
发布者: CN=zhong, OU=Unknown, O=Unknown, L=changping, ST=beijing, C=cn
序列号: 703dbee9
有效期为 Fri Apr 03 08:33:19 CST 2020 至 Tue Aug 20 08:33:19 CST 2047
证书指纹:
    MD5: OC:BE:13:D1:15:36:FF:69:F6:76:84:E2:B8:5B:85:28
    SHA1: BE:95:31:9F:9B:91:D1:02:B7:D8:FE:4B:3C:57:51:CC:F4:7C:7C
    SHA256: 1F:7B:5B:7A:32:D0:04:59:3D:4C:27:2F:33:7C:15:97:72:B5:71:87:F7:13:40:F5:59:12:E3:62:0E:59:2F:04
签名算法名称: SHA256withRSA
主体公钥算法: 2048 位 RSA 密钥
版本: 3

扩展:
#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 84 1A CE 2C B5 E5 DD F0 20 59 B8 AB 7C A6 03 C2 ..... Y.....
0010: 2B 9E 61 17                                     +.a.
]
]

*****
*****

Warning:
JKS 密钥库使用专用格式。建议使用 "keytool -importkeystore -srckeystore key.jks -destkeystore key.jks -deststoretype pkcs12" 迁移到行业标准格式 PKCS12。

```

## 2.3 配置打包 app 的网络请求

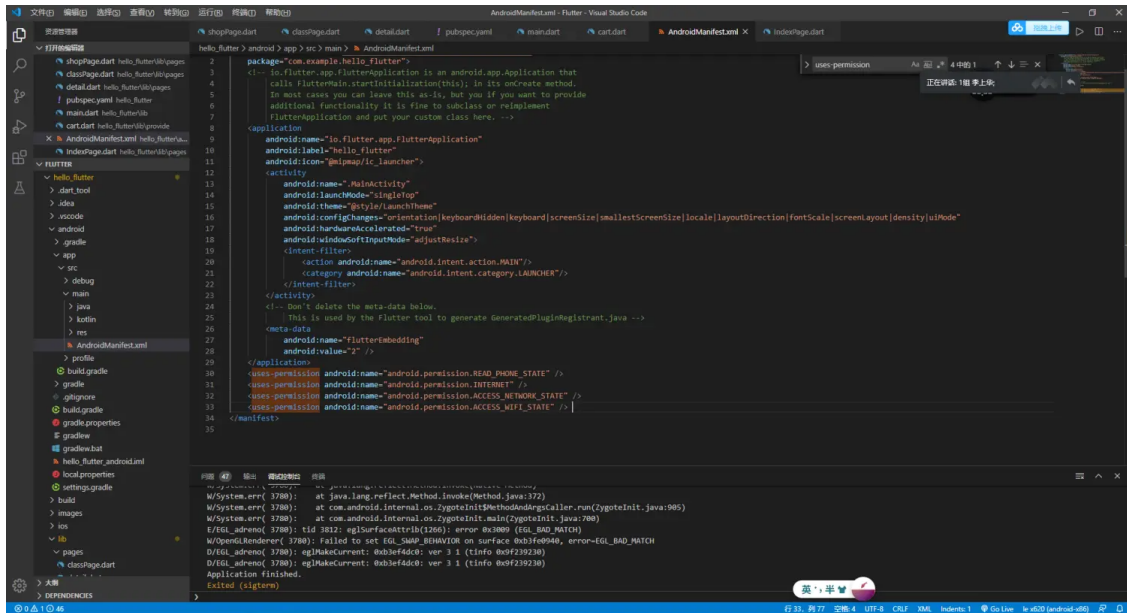
在这个文件里，  
android\app\src\profile\AndroidManifest.xml.manifest 在当前目录文件添加

```

<uses-permission
android:name="android.permission.READ_PHONE_STATE" />
<uses-permission
android:name="android.permission.INTERNET" />
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE"
/>
<uses-permission
android:name="android.permission.ACCESS_WIFI_STATE" />
//*****在application和manifest结束标签之前添加

```

如图



## 启用混淆配置

默认情况下，Flutter不会混淆和压缩Android原生代码，当项目使用的第三方依赖库需要添加混淆配置时，需要添加Flutter相关类的禁止混淆策略。

## 三、配置混淆文件

在android/app下创建proguard-rules.pro文件，并添加以下规则：

```
## Flutter相关类
-keep class io.flutter.app.** {*; }
-keep class io.flutter.plugin.** {*; }
-keep class io.flutter.util.** {*; }
-keep class io.flutter.view.** {*; }
-keep class io.flutter.** {*; }
-keep class io.flutter.plugins.** {*; }
```

上面这些配置会保护Flutter引擎类库不会混淆。

## 四 启动混淆和压缩

编辑 /android/app/build.gradle 文件，在 release 编译类型下添加混淆和压缩配置

```
buildTypes {
    release {
        // TODO: Add your own signing config for the release build.
        // Signing with the debug keys for now, so `flutter run --release` works.
        signingConfig signingConfigs.release

        minifyEnabled true
        useProguard true
        proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
    }
}
```

## 添加完了这个项目之后 Android 应用打包

使用命令行:

cd <app dir> (<app dir> 为您的工程根目录).

运行 flutter build apk (flutter build 默认会包含 --release 选项).

打包好的发布 APK 位于 <app dir>build/app/outputs/apk/app-release.apk

```
D:\day8project_code\Flutter\hello_flutter>flutter build apk
You are building a fat APK that includes binaries for android-arm, android-arm64, android-x64.
If you are deploying the app to the Play Store, it's recommended to use app bundles or split the APK to reduce the APK size.
To generate an app bundle, run:
  flutter build appbundle --target-platform android-arm,android-arm64,android-x64
Learn more on: https://developer.android.com/guide/app-bundle
To split the APKs per ABI, run:
  flutter build apk --target-platform android-arm,android-arm64,android-x64 --split-per-abi
Learn more on: https://developer.android.com/studio/build/configure-apk-splits#configure-abi-split
Removed unused resources: Binary resource data reduced from 46KB to 37KB: Removed 20%
Running Gradle task 'assembleRelease'...
Running Gradle task 'assembleRelease'... Done                    41.6s
✓ Built build/app/outputs/apk/release/app-release.apk (18.4MB).

D:\day8project_code\Flutter\hello_flutter>
```