# 打包发布

### 生成一个签名密钥

官网：<https://reactnative.cn/docs/signed-apk-android>

你可以用keytool命令生成一个私有密钥。在 Windows 上keytool命令放在 JDK 的 bin 目录中（比如C:\Program Files\Java\jdkx.x.x\_x\bin），你可能需要在命令行中先进入那个目录才能执行此命令。

$ keytool -genkeypair -v -storetype PKCS12 -keystore my-release-key.keystore -alias my-key-alias -keyalg RSA -keysize 2048 -validity 10000

这条命令会要求你输入密钥库（keystore）和对应密钥的密码，然后设置一些发行相关的信息。最后它会生成一个叫做my-release-key.keystore的密钥库文件。

在运行上面这条语句之后，密钥库里应该已经生成了一个单独的密钥，有效期为 10000 天。--alias 参数后面的别名是你将来为应用签名时所需要用到的，所以记得记录这个别名。

**注意：请记得妥善地保管好你的密钥库文件，一般不要上传到版本库或者其它的地方。**

### 设置 gradle 变量

1. 把my-release-key.keystore文件放到你工程中的android/app文件夹下。
2. 编辑~/.gradle/gradle.properties（全局配置，对所有项目有效）或是项目目录/android/gradle.properties（项目配置，只对所在项目有效）。如果没有gradle.properties文件你就自己创建一个，添加如下的代码（注意把其中的\*\*\*\*替换为相应密码）

**注意：~符号表示用户目录，比如 windows 上可能是**C:\Users\用户名**，而 mac 上可能是**/Users/用户名**。**

MYAPP\_RELEASE\_STORE\_FILE=my-release-key.keystore  
MYAPP\_RELEASE\_KEY\_ALIAS=my-key-alias  
MYAPP\_RELEASE\_STORE\_PASSWORD=\*\*\*\*\*  
MYAPP\_RELEASE\_KEY\_PASSWORD=\*\*\*\*\*

上面的这些会作为 gradle 的变量，在后面的步骤中可以用来给应用签名。

**关于密钥库的注意事项:**：如果你不想以明文方式保存密码，同时你使用的是 macOS 系统，那么你也可以把密码[保存到钥匙串（Keychain）中](https://pilloxa.gitlab.io/posts/safer-passwords-in-gradle/" \t "/Users/zhuyuhui/Documents\\x/_blank)。这样一来你就可以省略掉上面配置中的后两行（即 MYAPP\_RELEASE\_STORE\_PASSWORD 和 MYAPP\_RELEASE\_KEY\_PASSWORD）。

### 把签名配置加入到项目的 gradle 配置中

编辑你项目目录下的android/app/build.gradle，添加如下的签名配置：  
android {  
 ...  
 defaultConfig { ... }  
 signingConfigs {  
 release {  
 if (project.hasProperty('MYAPP\_RELEASE\_STORE\_FILE')) {  
 storeFile file(MYAPP\_RELEASE\_STORE\_FILE)  
 storePassword MYAPP\_RELEASE\_STORE\_PASSWORD  
 keyAlias MYAPP\_RELEASE\_KEY\_ALIAS  
 keyPassword MYAPP\_RELEASE\_KEY\_PASSWORD  
 }  
 }  
 }  
 buildTypes {  
 release {  
 ...  
 signingConfig signingConfigs.release  
 }  
 }  
}  
...

### 生成发行 APK 包