# **Deploying flutter apps to the PlayStore**

## **1. Create a developer account**

Access the Google Play Console and create a new developer account, for that is charged a single payment of US$ 25, after that we will be able to publish our app.

## **2. Update the app name**

Open AndroidManifest.xml (*android/app/src/main/AndroidManifest.xml*) and update the android label. The android label represents the application name, it’s the one that will be displayed on our phone as the app name.

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="com.myapp.name">

<application

**android:label="My App Name"**

android:name="${applicationName}"

android:icon="@mipmap/ic\_launcher">

## **3. Generate a keystore**

To be able to deploy the app to the store we’ll need to sign it, the first thing we should do is create a keystore.

To generate a new keystore run the following command:

**Windows**

keytool -genkey -v -keystore C:\Users\USER\_NAME\my-key.jks -storetype JKS -keyalg RSA -keysize 2048 -validity 10000 -alias my-key-alias

**Linux/Mac**

keytool -genkey -v -keystore ~/my-key.jks -keyalg RSA -keysize 2048 -validity 10000 -alias my-key-alias

When creating the new keystore we will be asked to enter a password, it’s important to keep it, since it will be necessary later.

## **4. Move the key file to the project**

Move the key file (my-key.jks in the exemple) to android/app, don’t forget to add the keystore into .gitignore if we have a public repository.

## **5. Create a key.properties file**

We’ll create a file that references the keystore, to do that, inside the android folder create a ‘key.properties’ file. Open the file and paste the content below, updating it with your keystore values.

storePassword=123456

keyPassword=123456

keyAlias=my-key-alias

storeFile=my-key.jks

## **6. Updating build.gradle**

Then go to **android/app/build.gradle** and update it with the highlighted lines:

apply plugin: 'com.android.application'

apply plugin: 'kotlin-android'

apply from: "$flutterRoot/packages/flutter\_tools/gradle/flutter.gradle"

**def keystoreProperties = new Properties()**

**def keystorePropertiesFile = rootProject.file('key.properties')**

**if (keystorePropertiesFile.exists()) {**

**keystoreProperties.load(new FileInputStream(keystorePropertiesFile))**

**}**

android {

compileSdkVersion flutter.compileSdkVersion compileOptions {

sourceCompatibility JavaVersion.VERSION\_1\_8

targetCompatibility JavaVersion.VERSION\_1\_8

} kotlinOptions {

jvmTarget = '1.8'

} sourceSets {

main.java.srcDirs += 'src/main/kotlin'

} defaultConfig {

// TODO: Specify your own unique Application ID (https://developer.android.com/studio/build/application-id.html).

applicationId "com.deployapp.test.deploy\_app\_test"

minSdkVersion flutter.minSdkVersion

targetSdkVersion flutter.targetSdkVersion

versionCode flutterVersionCode.toInteger()

versionName flutterVersionName

} **signingConfigs {**

**release {**

**keyAlias keystoreProperties['keyAlias']**

**keyPassword keystoreProperties['keyPassword']**

**storeFile keystoreProperties['storeFile'] ? file(keystoreProperties['storeFile']) : null**

**storePassword keystoreProperties['storePassword']**

**}**

**}** buildTypes {

release {

// signingConfig signingConfigs.debug

**signingConfig signingConfigs.release**

}

}

}

## **7. Build the app**

Now we can build the app as a appbundle, so it can be deployed to the store.

Run in your terminal:

flutter build appbundle

If you are using flavors run:

flutter build appbundle –flavor production

And if you created you app using very good ventures run:

flutter build appbundle -t lib/main\_production.dart –flavor production

After the build has finished, a new message will appear in the terminal, showing the output folder where your .aab file is located.

## **8. Deploy the app**

To deploy the application we need to create a new app at Google Play Console:

Go to “**Internal testing**”, create a new release, and upload our .aab

Then add testers to the “internal testing” release, creating a email list

## **9. Updating the app**

To update the app we need to provide a version name and version code. To do that go to pubspec.yaml.

**A.B.C**+X: Represents the version name (Visible to the public).  
A.B.C+**X**: Represents the version code, it has to be an integer (Not Visible to the public).

Update your version name and code before each build.

## **10. Backup your keystore**

To be able to update our app we need to keep the keystore in our project, so that if we lose it, we willl not be able to update our app anymore. Having that in mind, it’s a good idea to do a backup of it.