Google Play Store Publishing

1/26/16

Bryan Smith

Contents

Google Play Store Publishing 1

Contents 2

Purpose 3

Key points 3

Setup Google Developer Account 3

Invite Students 3

Keystore 4

- Considerations 4
- Security 5

References 5

Purpose

This document will provide an overview of setting up a Google Developer Account in order to allow the Department of Math and Computer Science of the University of Virginia's College at Wise to publish Android applications on the Google Play Store. Henceforth, the Department of Math and Computer Science of the University of Virginia's College at Wise will be referred to simply as the College. Most of the content of this document is from Google from the locations provided in the references at the end of this document. However, some customizations or omissions were made for the College's specific situation.

Key points

- One time $25 dollar registration fee.
- Create main account and invite students who need access to publish (Only one from each group really needed, or none if admin will publish for them).
- Keep and secure a single keystore recommended for ease of use.

Setup Google Developer Account

This will require a Google Account. It is recommended that a new one be made for the College if one does not already exist.

1. Visit the Google Play Developer Console.

2. Enter basic information about your **developer identity** — name, email address, and so on. You can modify this information later.

3. Read and accept the **Developer Distribution Agreement** for your country or region. Note that apps and store listings that you publish on Google Play must comply with the Developer Program Policies and US export law.

4. Pay a **$25 USD registration fee** using Google payments. If you don't have a Google payments account, you can quickly set one up during the process.

5. When your registration is verified, you'll be notified at the email address you entered during registration.

Invite Students

If students need access in order to publish, you do not need to give them the account credentials to the developer account. You can invite them to have access using their own Google Account. This allows you to manage who has access or not.

1. Sign in to your Google Play Developer Console at https://play.google.com/apps/publish/.

2. Go to Settings

3. Under User Accounts & Rights, click Invite New User

Keystore

A **keystore** is a binary file that contains a set of private keys. You must keep your keystore in a safe and secure place. A **private key** represents the entity to be identified with the app, such as a person or a company. The College may opt to let each project group create and handle their own keystore. If this is done however, the apps may not be able to be upgraded later on once those students are no longer at the College. Note you can store more than one private key in a keystore. Meaning you could have a separate private key for each app or project you have. However, I personally recommend to use a single keystore and a single private key for ALL apps that will be published by the College. It is up the College in order to determine the best route to take and how they want to handle the environment and data.

Considerations

You should sign all of your apps with the same certificate throughout the expected lifespan of your applications. There are several reasons why you should do so:

- App upgrade: When the system is installing an update to an app, it compares the certificate(s) in the new version with those in the existing version. The system allows the update if the certificates match. If you sign the new version with a different certificate, you must assign a different package name to the application—in this case, the user installs the new version as a completely new application.

- App modularity: Android allows apps signed by the same certificate to run in the same process, if the applications so requests, so that the system treats them as a single application. In this way you can deploy your app in modules, and users can update each of the modules independently.

- Code/data sharing through permissions: Android provides signature-based permissions enforcement, so that an app can expose functionality to another app that is signed with a specified certificate. By signing multiple apps with the same certificate and using signature-based permissions checks, your apps can share code and data in a secure manner.

- Some APIs (especially Google APIs) grant permissions to use the API based on the signing certificate of the app. So if an app if signed with a keystore other than what it was setup for in the API, then the app may no longer work as it will likely no longer have access to the API

If you plan to support upgrades for an app, ensure that your key has a validity period that exceeds the expected lifespan of that app. A validity period of 25 years or more is recommended. When your key's validity period expires, users will no longer be able to seamlessly upgrade to new versions of your application.

If you plan to publish your apps on Google Play, the key you use to sign these apps must have a validity period ending after 22 October 2033. Google Play enforces this requirement to ensure that users can seamlessly upgrade apps when new versions are available.

Security

Maintaining the security of your private key is of critical importance, both to you and to the user. If you allow someone to use your key, or if you leave your keystore and passwords in an unsecured location such that a third-party could find and use them, your authoring identity and the trust of the user are compromised.

If a third party should manage to take your key without your knowledge or permission, that person could sign and distribute apps that maliciously replace your authentic apps or corrupt them. Such a person could also sign and distribute apps under your identity that attack other apps or the system itself, or corrupt or steal user data.

Your private key is required for signing all future versions of your app. ***If you lose or misplace your key, you will not be able to publish updates to your existing apps. You cannot regenerate a previously generated key.***

Your reputation as a developer entity depends on your securing your private key properly, at all times, until the key is expired. Here are some tips for keeping your key secure:

- Select strong passwords for the keystore and key.
- Do not give or lend anyone your private key, and do not let unauthorized persons know your keystore and key passwords.
- Keep the keystore file containing your private key in a safe, secure place.

In general, if you follow common-sense precautions when generating, using, and storing your key, it will remain secure.

References

- http://developer.android.com/distribute/googleplay/start.html
- http://developer.android.com/tools/publishing/app-signing.html