Discussion-8 Prepare an Android App for Deployment

Discussion Topic:

What are the necessary steps to prepare an Android app for deployment to the Google Play Store?

My Post:

Hello Class,

For this discussion, I chose the following topic: What are the necessary steps to prepare an Android app for deployment to the Google Play Store?

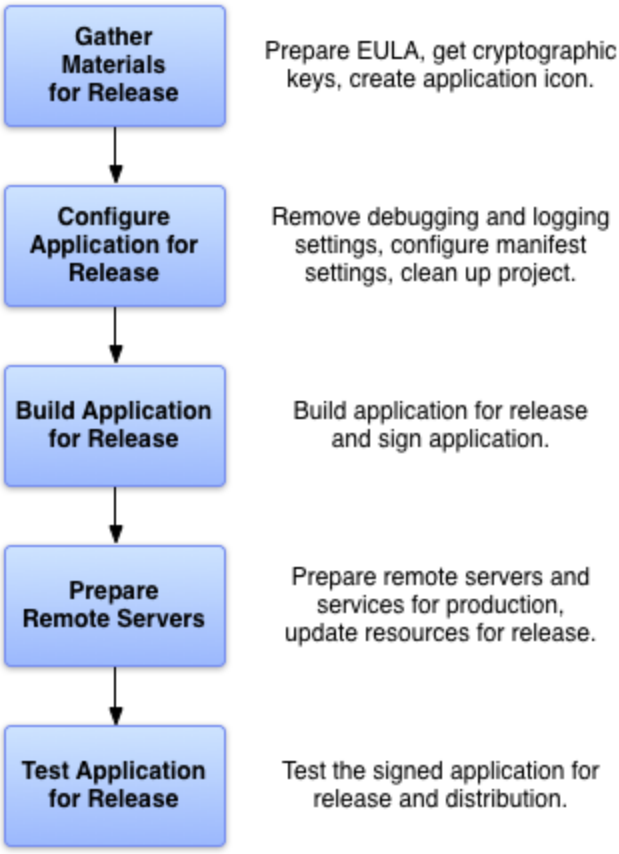
The Google Play Store is the primary channel through which Android developers distribute their apps. It is an ecosystem that allows them to publish, manage, monetize, and promote their apps globally. To successfully release the app and publish it in the store, the developer should prepare it to be deployed; this is merely a suggestion, but it is more like a requirement. To prepare an app for release, the developer needs to configure, build, and test a release version of the app (Android Developers, n.d.a). This post explores the tasks required to prepare an Android app for release.

Android Package Kit (APK)

Before preparing an Android app for release, it is important to understand what the Android Package Kit (APK) is. An APK file contains all the necessary components to run an app on an Android device (Gillis, n.d.). It Includes:

- **AndroidManifest.xml.** An Android manifest that describes the name, version, access rights, library and other contents of the APK file.
- **assets/.** App assets and resource files.
- **classes.dex.** Compiled Java classes in the DEX file format run on the device.
- **lib/.** Contains platform-dependent compiled code and native libraries for device-specific architectures, such as x86 or x86_64.
- **META-INF/.** Contains the app certificate, manifest file, signature, and a list of resources.
- **res/.** Contains resources; for example, images that are not already compiled into resources.arsc.
- **resources.arsc.** Contains pre-compiled resources used by the app.

Figure 1
Android App Release Tasks



Note: The figure illustrates the main tasks to prepare your app for release. From "Prepare your app for release" by Android Developers (n.d.a).

Release Tasks

As shown in Figure 1, the tasks are a set of chronological steps that need to be followed to properly release an Android app; the task set includes gathering materials, configuring the app, building an APK or Android App Bundles (AAB) file, preparing remote servers, and testing the app for release.

Gathering Materials

The first task of preparing an app for release is to gather the essential materials and resources. This needs to be done before configuring the app, and it requires preparing an End-User License Agreement (EULA), getting a cryptographic key, and having an App icon (Android Developers, n.d.a).

- EULA is a mandatory requirement for publishing on the Google Play Store. It is a legal contract between the developer and the users. It describes the terms and conditions of use.
  The following link is a generic example of an EULA made for the Steelcase company: Mobile Application End User License Agreement by Steelcase (2017). Note, this is just an example, and should be used as an example, not as a legal EULA document for your app.
- A cryptographic key is required for publishing an Android app. The key is an app unique digital signature that is used to verify the identity of the app's author. It is a security measure that is

used to verify that the app has not been tampered with or corrupted since it was signed. This key is mandated by the Google Play Store, and without it, apps cannot be installed or updated on Android devices.
- App icons are also required as they are used to launch the app on Android devices. “A launcher icon is a graphic that represents your application. Launcher icons are used by Launcher applications and appear on the user’s Home screen” (Stuff MIT, n.d.).

App Configuration for Release

The second step for preparing an Android app for release is to configure the app for deployment. This includes.

- An app ID, which is the package name that will be used over the life of the app (Stuff MIT, n.d.). For example*, ‘com.example.myapp*’, which usually follows the reverse domain name convention, this ID will be used to update, identify, and integrate the app with another service. Note that once the ID is set, after the app is distributed, the ID cannot be changed.
- Turning off debugging and logging (if used), it is essential to deactivate logging and disable the debugging option before building an app for deployment. To deactivate logging, the *Log* methods in the app’s source files need to be disabled. To disable debugging *android:debuggable* attribute needs to be removed from the *<application>* tag in the app’s manifest file, or by setting the *android:debuggable* attribute to false in the manifest file. Additionally, any log files or static test files need to be removed. If used, any debugging tracing calls added code, such as *startMethodTracing()* and *stopMethodTracing()* need to be removed.
- It is also important to clean up the app’s project directories from stray and orphaned files by reviewing jni/, lib/, and src/ directories. For example, the jni/ directory needs to contain only source files associated with the Android NDK. The lib/ directory needs to contain only third-party library files or private library files. The src/ directory needs to contain only the source files for your application (.java and .aidl files). The src/ directory should not contain any .jar files. Additionally, private or proprietary data files that the app does not use need to be removed. For example, in the res/ directory, old drawable files, layout files, and values files that are no longer used by the app need to be removed. In the lib/ directory, test libraries need to be removed. In the assets/ directory and res/raw/ directory, check for raw asset files and static files that may need to be updated or removed before release.

Build the App for Release

The next task is to build the APK file, also called a release-ready APK file, or Android App Bundle (AAB) file; both file formats should be signed and optimized (Android Developers, n.d.a). To sign the app for release to the Google Play Store means that the app needs to possess a unique cryptographic key. The cryptographic key was addressed previously in the Gathering Materials section of this post. Google strongly recommends using the AAB file format for new applications. An AAB file is a publishing format that includes all your app's compiled code and resources; it also defers APK generation and signing to Google Play (Android Developers, n.d.b). Note that when using Android Studio or the Gradle build system from the command line, the build process is entirely automated. To build an Android app release

with Android Studio, the following YouTube video is a good source of information How to Create Singed AAB file in Android Studio (2023 Update) by The Code City (2023). The following YouTube the entire process of deploying an Android app on Google Play Store: How to Publish an Android App to Google Play | Updated 2024 by MJSD Coding (2024).

Prepare Severs

This task is relevant to the app release if the app relies on any external servers or services for its core functionality (Android Developers, n.d.a). Preparing the server for release involves verifying that the servers are secure, that they can handle the expected number of users (user load), and are configured for production use. This applies to apps using API calls to fetch data from a source that is not controlled by the app developer.

Testing For Release

The final task is testing of the app release build. This is done by testing it on the device and network conditions. Ideally, the app UI should at least be capable of handling one handset-sized device and one tablet-sized device. Google recommends using its Firebase Test Lab platform (Firebase, n.d.) to test Android apps across a variety of different devices and Android OS versions.

Summary

Preparing an Android app for release on the Google require a set of multiple steps, these tasks include understanding the structure of an APK and AAB file formats, gathering essential materials like a cryptographic key, EULA, and app icon; configuring the app by setting a unique ID, disabling debugging/logging, and cleaning project directories; and finally, building a signed and optimized release version, preferably as an AAB file format. Following these preparation tasks are essential for a smooth and successful app launch.

-Alex

References:

Android Developers (n.d.a). *Prepare your app for release.* Android. https://developer.android.com/studio/publish/preparing

Android Developers (n.d.b*). About Android App bundles*. Android. https://developer.android.com/guide/app-bundle

Firebase (n.d.). *Firebase Test Lab.* Google. https://firebase.google.com/docs/test-lab

Gillis, A. S. (n.d.). *APK file (Android Package Kit file format).* Tech Target Networks. https://www.techtarget.com/whatis/definition/APK-file-Android-Package-Kit-file-format#:~:text=Contents%20of%20an%20Android%20Package,resources%20used%20by%20the%20app.

MJSD Coding (2024, September 3). *How to publish an Android app to Google Play | Updated 2024* [Video]. YouTube. https://www.youtube.com/watch?v=d8uEdeMgikU

Steelcase (2017). Mobile application end user license agreement [PDF]. Steelcase. https://www.steelcase.com/content/uploads/2018/01/mobile_app_eula_ssa_19oct2017.pdf

Stuff MIT (n.d.). *Launcher icons*. Massachusetts Institute of Technology. https://stuff.mit.edu/afs/sipb/project/android/docs/guide/practices/ui_guidelines/icon_design_launcher.html#

The Code City (2023, August 2) *How to create singed AAB file in Android Studio (2023 Update)* [Video]. YouTube. https://www.youtube.com/watch?v=qMAtgMP0xyg