Android Deployment Play Store

v3.0

13/01/2023

Introducción

En el presente documento, se presenta el primer paso para desplegar nuestra aplicación en un dispositivo móvil con sistema operativo Android y consideraciones de seguridad. El segundo paso se abordará posteriormente.

En versiones anteriores de Ionic (actualmente v6), se entregaba un paso a paso para realizar la firma manual de nuestra aplicación. Lo que implicaba hacer una serie de instalaciones y configuraciones sobre el PC del desarrollador.

Hoy en día lo que se recomienda es instalar Android Studio y firmar la aplicación desde las opciones que te ofrece la aplicación. Pero daremos un vistazo a algunos conceptos importantes.

Términos, Abreviaturas y Acrónimos

1. APK: Paquete para sistemas operativos Android, variante del formato JAR de Java

Antes de Comenzar

Esta guía cubre cómo ejecutar y depurar aplicaciones Ionic en emuladores y dispositivos de Android usando Capacitor.

Android Studio es el IDE para crear aplicaciones nativas de Android. Incluye el SDK de Android, que deberá configurarse para su uso en la línea de comandos.

Android Studio también se usa para crear dispositivos virtuales de Android, que son necesarios para el emulador de Android. Las aplicaciones Ionic también se pueden iniciar en dichos dispositivos virtuales.

Una vez instalado, abre Android Studio. El IDE debe detectar que es necesario instalar el SDK de Android. En la pantalla de configuración de los componentes del SDK, finalice la instalación del SDK. Tome nota de la ubicación del SDK de Android.

De manera predeterminada, se instala la plataforma SDK estable más reciente, que incluye una colección de paquetes necesarios para orientar esa versión de Android.

Para instalar imágenes del sistema y otros paquetes menores de la plataforma SDK, es posible que deba asegurarse de que seleccionar la opción “Mostrar detalles del paquete” en la parte inferior del Administrador de SDK.

El SDK de Android incluye útiles herramientas de línea de comandos. Antes de que se puedan usar, se deben establecer algunas variables de entorno. Te recomiendo consultar la documentación sobre la configuración y la persistencia de variables de entorno en sesiones de terminal https://ionicframework.com/docs/developing/android#java.

Generar APK (Sin Firmar)

El primer paso es generar el APK mediante

npx cap add [options]

npx cap sync

ionic capacitor build [options]

*options: android ó ios

NOTA: Esta puede usarse en un dispositivo

Una opción es abrir el proyecto en Android Studio y desde ahí obtener el APK:

npx cap open android

También puedes abrir la APK sin firmar directo en el celular sin pasar por Android Studio (debes tener instalado el complemento Ionic en VSC)

ionic cap run android --l --external

Para errores del tipo: **Could not find an installer version of Gradle either in Android Studio** deben descargar la última versión siguiendo estas instrucciones.

Si tienes un error como este Could not find tools.jar. Please check that C:\Program Files\Java\jre1.8.0_151 contains a valid JDK installation sigue estas instrucciones.

NOTA: la APK generada para ser usada y/o distribuida necesita ser firmada.

Plataforma Generada

Cuando ejecutas los comandos anteriores de adicionar, sincronizar y compilar en una plataforma, se produce una serie de creaciones de objetos como la carpeta platforms, plugins y resources como también el archivo config.xml

● Platform: Esta carpeta contiene los proyectos híbridos generados por Ionic en este caso la plataforma es Android y se estructura de la misma forma que un proyecto Android Nativo

● plugins: Esta carpeta contiene todos los plugins que necesarios y que se han instalado aparte durante el desarrollo del proyecto Ionic Angular

● resources: Contiene los recursos gráficos de la aplicación para las distintas plataformas

● config.xml: Archivo de configuración para los datos básicos de la aplicación

Archivo config.xml

Este archivo generado al momento de construir una aplicación de release para android, contiene ciertas configuraciones como el nombre `<name>MyApp</name>` del proyecto que será el nombre visible en el cajón de aplicaciones. Como también el id y la versión de la aplicación `<widget id="io.ionic.starter" version="0.0.1" .../>` donde el id tiene un rol muy importante ya que será el identificador único de la aplicación y la versión va cambiando según los update que se le realizan.

Por último, se tiene el android-minSdkVersion `<preference name="android minSdkVersion" value="19" />` en el cual se tiene que estipular el Android mínimo que acepta la aplicación, a continuación, hay una guía para saber escoger al API Level.

Esto influye directamente en el archivo AndroidManifest.xml un archivo muy importante en aplicaciones Android Nativas.

Seguridad

Para componentes donde se le permite al usuario cualquier contenido personalizado que puede ser texto sin formato o HTML debe considerarse no confiable.

Al igual que con cualquier entrada no confiable, es importante desinfectarla antes de realizar cualquier operación con esa entrada en particular cuando esa entrada puede ser algo con innerHTML ya que proporciona un punto de entrada para ataques Cross Site Scripting (XSS)

Ionic tiene integrado métodos básicos de desinfección para los componentes que proporciona, llámese ion-input o ion-alert, pero en los componentes desarrollados por el propio desarrollador es él que debe realizar esa tarea. Para esto último se recomienda usar sanitize-html

Ionic tiene un desinfectante incorporado que puede deshabilitarse de manera global pero no se recomienda ya que la aplicación puede volverse vulnerable a ataques XSS.

Si desea saltarse el Desinfectante incorporado por Ionic y realizarlo manualmente puede hacer uso de la función **IonicSafeString** el cual permite desinfectar cualquier String.

Generar APK (Fimada)

Existen 2 formas de poder generar la firma de nuestra aplicación (en realidad tres: dos manuales y una automática), siendo la primera mediante keytool y la segunda mediante Android Studio, la tercera no se revisará en este documento ya que se debe seguir el paso a paso de Google Play (y que está en constante actualización).

Tanto para la primera como la segunda, es necesario tener cierta configuración, por lo cual en esta sección se presentará la configuración que debes tener para poder firmar la aplicación de diferentes formas.

Generar Clave Manual con keytool

Si ya posees una clave, puedes saltarte este ítem.

Generamos una clave privada usando el comando keytool que viene con el SDK de Android

Paso 1: Moverse a la carpeta bin del JDK

```
cd 'C:\Program Files\Java\jdk1.8.0_261\bin'
```

Paso 2: Generar key

```
./keytool -genkey -v -keystore my-release-key.keystore -alias alias_name
-keyalg RSA -keysize 2048 -validity 10000
```

Donde **./keytool -genkey** es la herramienta que nos generar la key, **-v** nos da una salida detallada del proceso, **-keystore** es el nombre al almacén de claves donde estará almacenada nuestra key, **-alias** es el nombre de nuestra key, **-keyalg** nombre del algoritmo de la key, **-keysize** es el tamaño en bit de la clave y **-validity** es el número de días de validez del certificado.

Al ejecutar este comando lo primero que nos pedirá será la contraseña del almacén de claves, si es nueva pedirá repetir la contraseña. Luego pedirá su nombre y apellido, Nombre de la unidad de la organización (en este caso se puso personal), en organización se ingresó un nombre y apellido, posteriormente solicitará el nombre de la ciudad, estado o provincia (Region) y pais (CL).

```
Introduzca la contraseña del almacén de claves:
Volver a escribir la contraseña nueva:
¿Cuáles son su nombre y su apellido?
  [Unknown]:  IAN CARDENAS C
¿Cuál es el nombre de su unidad de organización?
  [Unknown]:  PERSONAL
¿Cuál es el nombre de su organización?
  [Unknown]:  IAN CARDENAS C
¿Cuál es el nombre de su ciudad o localidad?
  [Unknown]:  SANTIAGO
¿Cuál es el nombre de su estado o provincia?
  [Unknown]:  RM
¿Cuál es el código de país de dos letras de la unidad?
  [Unknown]:  CL
```

Luego de ingresar todos esos datos solicitara la confirmación ustedes ingresan **si**

```
¿Es correcto CN=IAN CARDENAS C, OU=PERSONAL, O=IAN CARDENAS C, L=SANTIAGO, ST=RM, C=CL?
  [no]:
```

Al final les pedira la contraseña para key_ionic_app que si es la misma que el almacén presionar "INTRO" si no la ingresan y presiona "INTRO"

Si obtienen el siguiente error (tiene una referencia de Acceso Denegado):

```
[Almacenando release-key.keystore]
error de herramienta de claves: java.io.FileNotFoundException: release-key.keystore (Acceso denegado)
java.io.FileNotFoundException: release-key.keystore (Acceso denegado)
        at java.io.FileOutputStream.open0(Native Method)
        at java.io.FileOutputStream.open(FileOutputStream.java:270)
        at java.io.FileOutputStream.<init>(FileOutputStream.java:213)
        at java.io.FileOutputStream.<init>(FileOutputStream.java:101)
        at sun.security.tools.keytool.Main.doCommands(Main.java:1196)
        at sun.security.tools.keytool.Main.run(Main.java:368)
        at sun.security.tools.keytool.Main.main(Main.java:361)
```

Es porque no tienen autorización de escritura en el disco C: por ende, lo cambiaremos a una unidad lógica que tengamos permisos de escritura

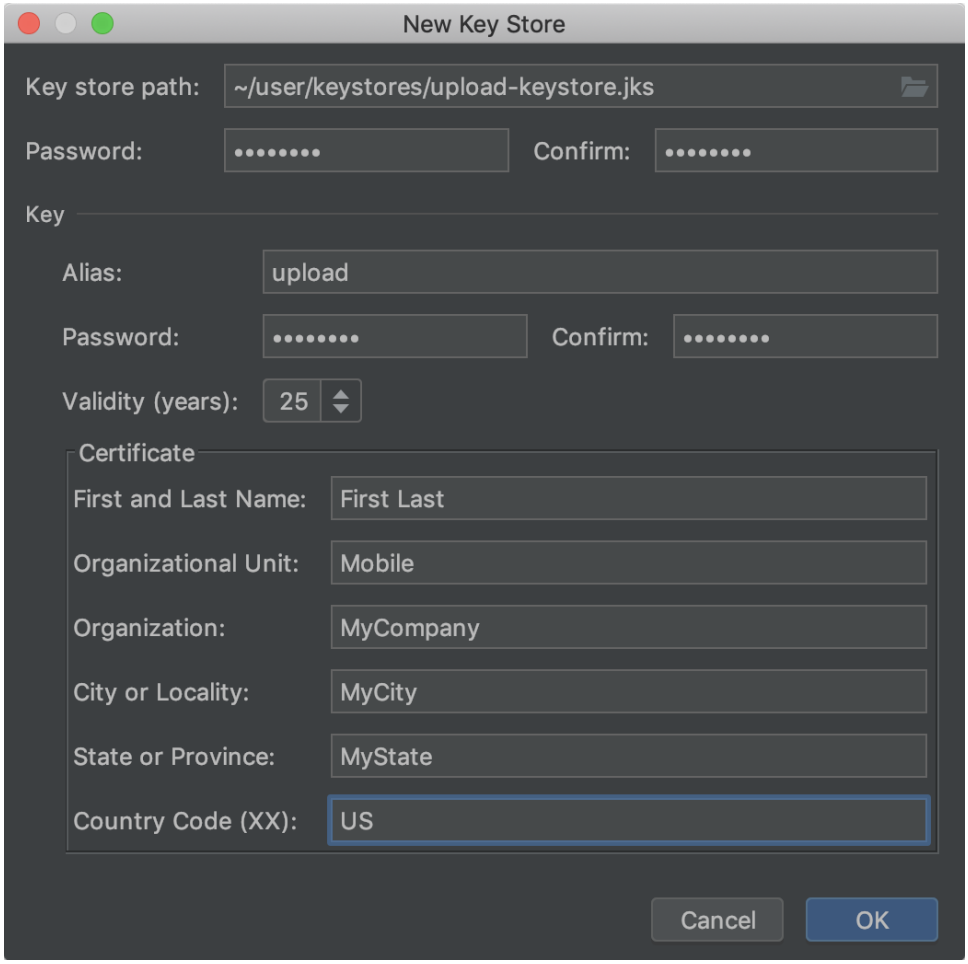
./keytool -genkey -v -keystore d:\release-key.keystore -alias key_ionic_app -keyalg RSA -keysize 2048 -validity 10000

Repiten el mismo procedimiento y deberían tener salida exitosa. Lo más importante es que tengas claro dónde quedó almacenada tu firma. Ya que el siguiente paso será asociar esta firma a tu aplicación.

Generar Clave Manual con Android Studio

Si todavía no tienes una clave, que sirve para configurar la firma de apps de Play, puedes generar una con Android Studio de la siguiente manera:

1. En la barra de menú, haz clic en **Build > Generate Signed Bundle/APK**
2. En el diálogo **Generate Signed Bundle or APK**, selecciona **Android App Bundle** o **APK**, y haz clic en **Next**.
3. En el campo **Key store path**, haz clic en **Create new**.
4. En la ventana **New Key Store**, proporciona la información solicitada para el almacén de claves y la clave.



5. Almacén de claves
   - **Key store path**: Selecciona la ubicación en la que quieras crear el almacén de claves. Además, se debe agregar un nombre de archivo al final de la ruta de acceso de ubicación con la extensión .jks.
   - **Password**: Crea y confirma una contraseña segura para el almacén de claves.
6. Clave
   - **Alias**: Ingresa un nombre de identificación para la clave.
   - **Password**: Crea y confirma una contraseña segura para tu clave. Debe ser la misma que la contraseña del almacén de claves.
   - **Validity** (years): Establece durante cuántos años será válida la clave. Debe ser válida durante al menos 25 años para que puedas firmar actualizaciones con la misma clave durante todo el ciclo de vida de la app.

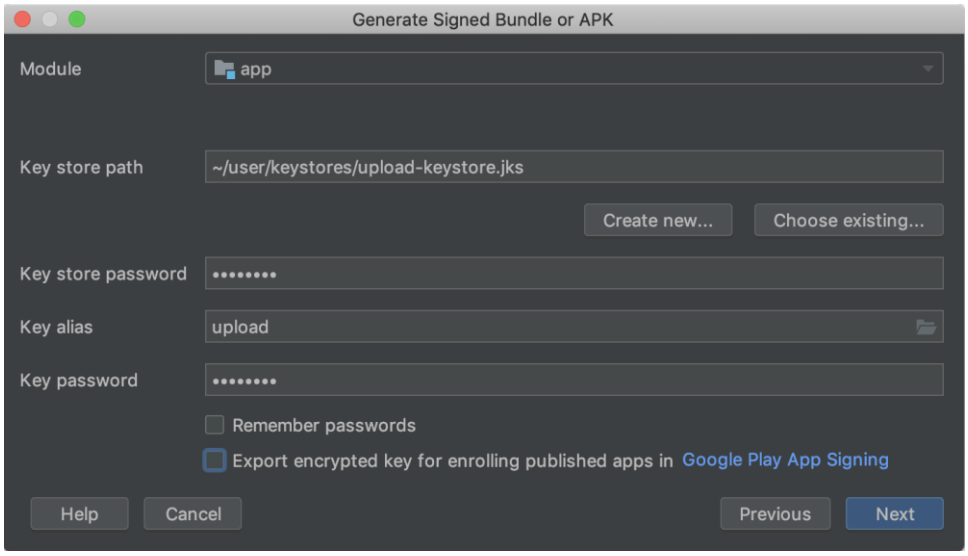
- **Certificate**: ingresa información acerca de ti para el certificado. Esta información no se muestra en la app, pero se incluye en el certificado como parte del APK.

Una vez que completes el formulario, haz clic en OK.

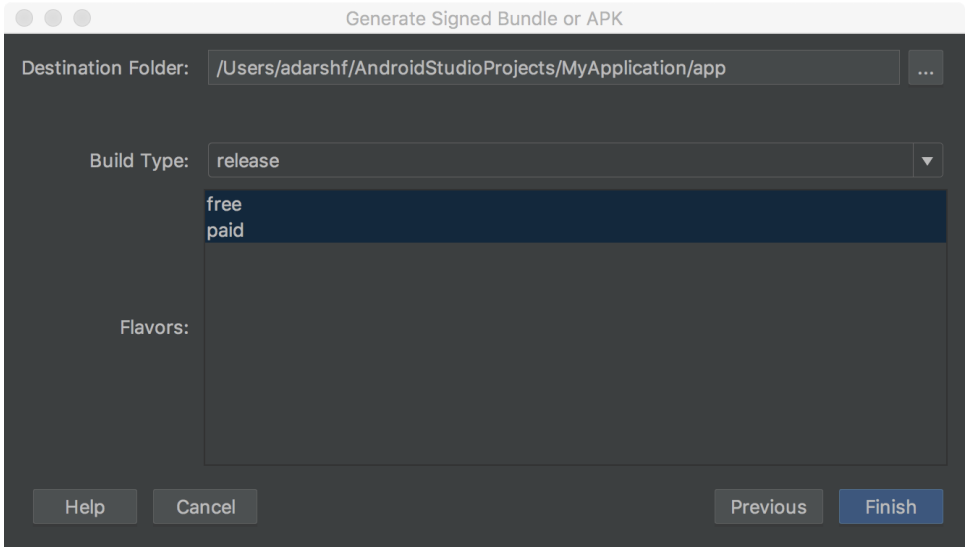
Firma de la Aplicación Manual(con llave de keytool o Android Studio)

Sigue los pasos que se muestran a continuación para firmar tu app con Android Studio y exportar una clave de firma de la app. Recuerda que lo primero que debes hacer es abrir tu aplicación en Android Studio desde Visual Studio Code con las líneas de comando indicadas al principio de este documento:

1. Si no tienes abierto el diálogo **Generate Signed Bundle or APK**, haz clic en **Build** > **Generate Signed Bundle/APK**.
2. En el cuadro de diálogo **Generate Signed Bundle or APK**, selecciona **Android App Bundle o APK**, y luego haz clic en **Next**.
3. Selecciona un módulo de la lista desplegable.
4. Especifica la ruta de acceso al almacén de claves, el alias de la clave y las contraseñas de ambos.
5. Si firmas un paquete de aplicación con una clave de firma de la app existente y más tarde quieres inscribir tu app en la firma de apps de Play, marca la casilla que aparece junto a **Export encrypted key** y especifica una ruta para guardar tu clave de firma como un archivo *.pepk encriptado. Luego, podrás usar la clave de firma de la app encriptada existente a fin de [inscribir una app existente en la firma de apps de Play].
6. Haz clic en Next.
7. En la siguiente ventana (que se muestra), selecciona una carpeta de destino para la app firmada, elige el tipo de compilación y, si corresponde, selecciona la variante de producto.



8. Si compilas y firmas un APK, deberás



seleccionar con qué versiones de firma quieres que sea compatible la app en **Signature Versions**. Para obtener más información, consulta los esquemas de firma de la app.

9. Haz clic en Finish.

IMPORTANTE: El proceso de publicación en Google Play necesita tu APK firmada manualmente, pero también te permite generar una firma Automática.