



Distribución del aplicativo Android

Release (Parte I)

Construir un Release de un aplicativo Android utilizando procedimiento de empaquetamiento para ser distribuido en entornos productivos.

- Unidad 1:
Acceso a datos en Android
- Unidad 2:
Consumo de API REST
- Unidad 3:
Testing
- Unidad 4:
Distribución del aplicativo Android



Te encuentras aquí



¿Qué aprenderás en esta sesión?

- *Configuraciones gradle para release*
- *Qué es la firma .jks*
- *Qué hacer en caso de que la firma se vea comprometida*

¿Sabes que es una firma digital?, ¿Sabías que las apps provenientes de la Google Play Store están firmadas digitalmente?



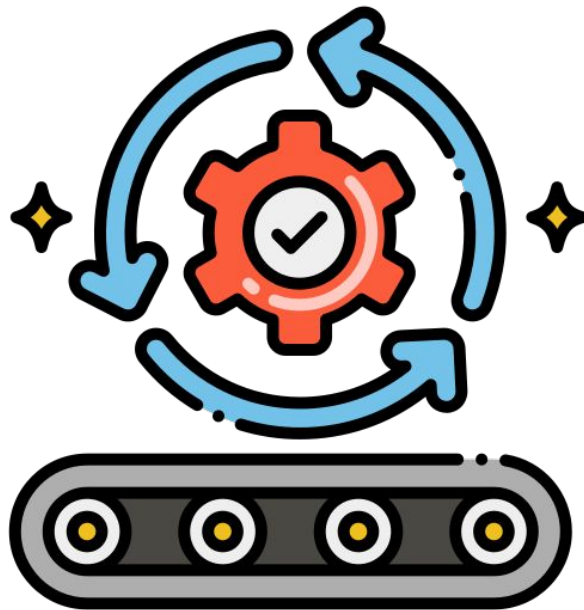
**/* Configuraciones gradle
para release */**

Configuraciones gradle para release

¿Qué es Gradle?

Recordemos que Gradle es una herramienta de automatización de compilación que se usa en proyectos de Android para automatizar el proceso de compilación, prueba e implementación de la aplicación.

Es responsable de administrar las dependencias del proyecto, compilar el código fuente y generar los archivos binarios de la aplicación.



Configuraciones gradle para release

Funciones

Y una de sus funciones es **Implementar la aplicación (Deploy)**.

Gradle se usa para implementar la aplicación en diferentes entornos, como ensayo, **producción** o versión beta.

Puede firmar los archivos binarios de la aplicación con los certificados necesarios y subirlos a las tiendas de aplicaciones u otros canales de distribución.



Configuraciones gradle para release

Para configurar tu archivo `build.gradle` para liberar una aplicación de Android, deberás realizar los siguientes pasos:

Especifica el número de versión de tu aplicación configurando las propiedades `versionCode` y `versionName` en su archivo `build.gradle`. Por ejemplo:

```
android {  
    defaultConfig {  
        versionCode 1  
        versionName "1.0"  
        // ...  
    }  
    // ...  
}
```

versionCode: es un valor tipo `Int` y no puede ser menor a la versión anterior

versionName: es un valor tipo `String`

Configuraciones gradle para release

Agrega el bloque `signatureConfigs` a tu archivo `build.gradle` para configurar las propiedades de firma de su aplicación. Esto es necesario para firmar tu aplicación con un almacén de claves para su lanzamiento. Por ejemplo:

```
android {  
    // ...  
    signingConfigs {  
        release {  
            keyAlias 'your_key_alias'  
            keyPassword 'your_key_password'  
            storeFile file('path/to/your_keystore.jks')  
            storePassword 'your_keystore_password'  
        }  
    }  
    // ...  
}
```

Configuraciones gradle para release

Establece el bloque `buildTypes` en tu archivo `build.gradle` para configurar las propiedades de compilación para tu aplicación. Esto es necesario para establecer la configuración de firma y otras opciones de compilación para la versión de lanzamiento. Por ejemplo:

```
android {  
    // ...  
    buildTypes {  
        release {  
            minifyEnabled true  
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'  
            signingConfig signingConfigs.release  
        }  
    }  
    // ...  
}
```

Configuraciones gradle para release

Finalmente, ejecute la tarea de ensamblarRelease para generar un archivo APK firmado para su aplicación. Puede encontrar el archivo APK generado en el directorio `app/build/outputs/apk/release`.

Por ejemplo, puede ejecutar el siguiente comando:

```
./gradlew assembleRelease
```

Ten cuenta que deberás reemplazar los valores:

- `your_key_alias`,
- `your_key_password`,
- `path/to/your_keystore.jks`
- `your_keystore_password`

Con sus propios valores.



Importante: Asegúrate de mantener su archivo de almacén de claves y sus contraseñas seguros y protegidos.

/* Qué es una firma digital */

¿Qué es una firma digital?

Una firma digital es una técnica matemática utilizada para verificar la autenticidad e integridad de un mensaje o documento digital. Es el equivalente digital de una firma o sello manuscrito.

Una firma digital se crea utilizando un algoritmo de firma digital, que utiliza una clave privada para cifrar un mensaje o documento. Esto produce una firma digital única que se puede adjuntar al mensaje o documento.



¿Qué es una firma digital?

Para verificar la firma digital, el destinatario utiliza la clave pública del remitente para descifrar la firma y confirmar que coincide con el mensaje o documento. Si se verifica la firma, proporciona seguridad de que el mensaje o documento no fue manipulado durante la transmisión y que se originó en el remitente.

Las firmas digitales se usan comúnmente en transacciones electrónicas, como banca en línea, comercio electrónico y documentos gubernamentales, para garantizar la autenticidad e integridad de la transacción. Proporcionan una forma segura y eficiente de autenticar y verificar documentos y mensajes digitales, y son un componente esencial de muchos sistemas de seguridad digital.

/* Qué es la firma .jks */

¿Qué es la firma .jks?



Un archivo JKS (Java KeyStore) es un tipo de archivo de almacén de claves que se utiliza en aplicaciones basadas en Java, incluidas las aplicaciones de Android. Un almacén de claves es una base de datos de certificados digitales y claves privadas que se utilizan para el cifrado y la autenticación.

En el desarrollo de Android, normalmente se usa un archivo JKS para firmar una aplicación antes de lanzarla a los usuarios. El archivo JKS contiene la clave privada y el certificado digital necesarios para firmar la aplicación, lo que garantiza que proviene de una fuente confiable y que no se ha manipulado.

Cómo se crea un archivo .jks

- Para generar un archivo JKS, puedes usar una herramienta como **keytool**, que se incluye con el Kit de desarrollo de Java (JDK). Puedes usar keytool para crear un nuevo archivo JKS, generar una nueva clave privada y un certificado digital, y agregarlos al almacén de claves.
- Al lanzar una aplicación de Android, puedes firmar la aplicación con la clave privada almacenada en el archivo JKS mediante la herramienta **jarsigner**. Esto garantiza que la aplicación sea auténtica y no se haya modificado desde que se firmó.
- Es importante mantener seguros el archivo JKS y la clave privada, ya que cualquier persona con acceso a la clave puede firmar y distribuir aplicaciones en su nombre. También se recomienda crear archivos JKS separados para diferentes aplicaciones o diferentes versiones de la misma aplicación, para reducir el riesgo de comprometer la clave privada.

Ejercicio

"Crea tu archivo JKS con keytool"



Crea tu archivo JKS con keytool

Para generar un archivo JKS con la herramienta **keytool** y usarlo para firmar una aplicación de Android, puede seguir estos pasos:

1. Abre un símbolo del sistema o una ventana de terminal (depende del SO que utilices).
2. Navega hasta el directorio donde deseas generar el archivo JKS.



Crea tu archivo JKS con keytool

3. Ejecuta el siguiente comando:

```
keytool -genkey -v -keystore myapp.keystore -alias myapp -keyalg RSA  
-keysize 2048 -validity 10000
```

Reemplace **myapp.keystore** con el nombre que desea dar a su archivo de almacén de claves y **myapp** con el nombre que desea dar al alias clave.

Listo, después de seguir estos pasos tendrás tu archivo de firmas, el cual usaremos más adelante.



```
usuario@machine ~ % keytool -genkey -v -keystore SuperAndroidApp.keystore -alias myapp -keyalg RSA  
-keysize 2048 -validity 10000
```

Enter keystore password: *****

Re-enter new password: *****

Enter the distinguished name. Provide a single dot (.) to leave a sub-component empty or press ENTER to use the **default** value in braces.

What is your first and last name?

[Unknown]: Usuario Firmas

What is the name of your organizational unit?

[Unknown]: Desarrollo

What is the name of your organization?

[Unknown]: Empresa de desarrollo

What is the name of your City or Locality?

[Unknown]: Santiago

What is the name of your State or Province?

[Unknown]: Santiago

What is the two-letter country code for this unit?

[Unknown]: CL

Is CN=Usuario Firmas, OU=Desarrollo, O=Empresa de desarrollo, L=Santiago, ST=Santiago, C=CL correct?

[no]: Yes

Generating 2,048 bit RSA key pair and self-signed certificate (SHA384withRSA) with a validity of 10,000 days

for: CN=ArchivoDeClaves1, OU=Desarrollo, O=Empresa de desarrollo, L=Santiago, ST=Santiago, C=CL

[Storing SuperAndroidApp.keystore]

{desafío}
latam_

```
usuario@machine ~ % ls -l | grep SuperAndroidApp
```

```
-rw-r--r-- 1 usuario staff 2790 Mar 28 06:56 SuperAndroidApp.keystore
```



Demostración

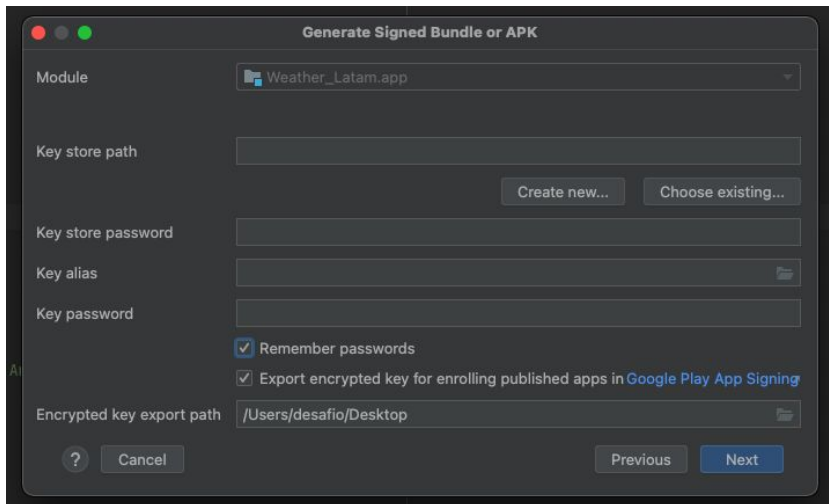
"Crea tu archivo JKS con Android Studio"



Crea tu archivo JKS con Android Studio

Asumiendo que aún no has creado tu archivo de firmas, puedes hacerlo utilizando el asistente de Android Studio, sigue estos pasos:

1. Menu > Build > **Generate Signed Bundle or APK**

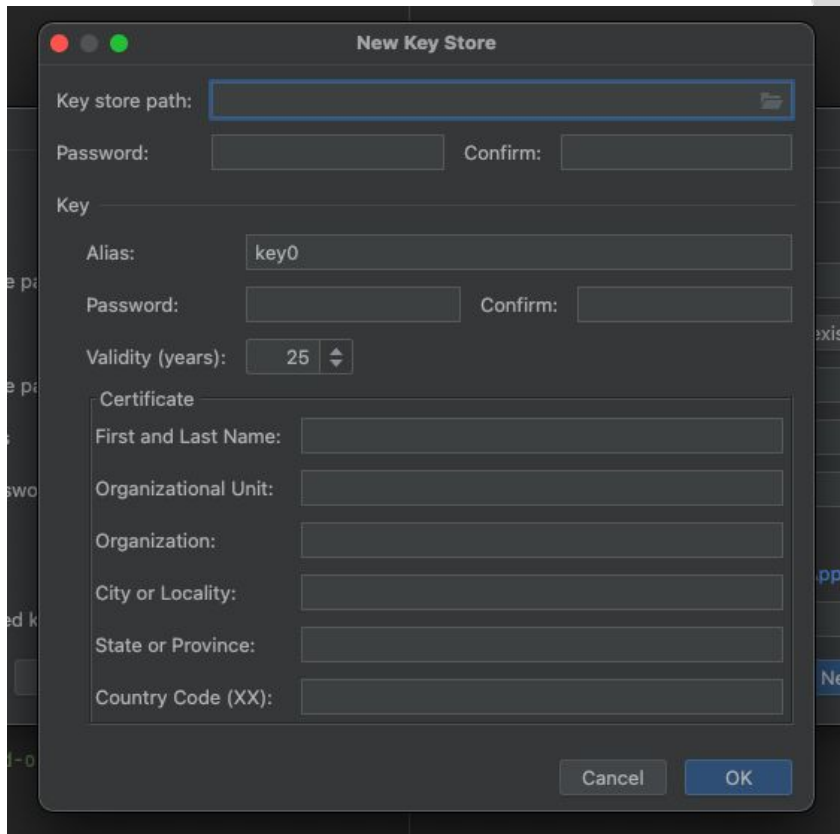


2. Luego presiona en “**Create new ...**”

{desafío}
latam_

Luego completa los datos

- **Key store path:** Es la ruta en la que se guardará el archivo de firmas.
- **Password:** contraseña del archivo JKS.
- **Alias:** nombre con el que se identifica la firma.
- **Password:** corresponde a la contraseña de la firma.
- **Validity:** indica el tiempo de duración de la firma, por defecto 25 años.
- **Certificate:** datos del certificado, pueden ser los datos del desarrollador



New Key Store

Key store path:

Password: Confirm:

Key

Alias:

Password: Confirm:

Validity (years):

Certificate

First and Last Name:

Organizational Unit:

Organization:

City or Locality:

State or Province:

Country Code (XX):

Cancel OK

/* Compromiso de la firma */

Qué hacer en caso de que la firma se vea comprometida

Algunos pasos a seguir

- **Comunícate con el soporte de Google Play Store:** si has publicado tu aplicación en Google Play Store, comunícate con el equipo de soporte para obtener ayuda. Es posible que puedan ayudarte a recuperar tu clave perdida o brindarte orientación sobre cómo proceder.
- **Genera una nueva clave:** si no puedes recuperar tu clave perdida, deberás generar una nueva clave y usarla para firmar futuras versiones de tu aplicación. Ten en cuenta que esto dará como resultado un nuevo certificado de firma, lo que puede causar algunos problemas a los usuarios que hayan instalado la versión anterior de su aplicación.

Qué hacer en caso de que la firma se vea comprometida

Algunos pasos a seguir

- **Haz una copia de seguridad de tu nueva clave:** una vez que hayas generado una nueva clave, asegúrate de hacer una copia de seguridad y guardarla en un lugar seguro. Esto ayudará a prevenir problemas similares en el futuro.

En general, es importante mantener su archivo JKS y tu clave privada seguros y respaldados, y tomar medidas para evitar la pérdida o el robo de su clave de firma.

Qué hacer en caso de que la firma se vea comprometida

Si pierdes tu archivo JKS o la clave privada asociada con él, ya no podrás firmar nuevas versiones de tu aplicación con la misma clave. Esto puede causar varios problemas, que incluyen:

1. **Incapacidad para actualizar su aplicación:** si pierde su archivo JKS, no podrá firmar nuevas versiones de su aplicación con la misma clave. Esto significa que no podrá publicar actualizaciones de su aplicación en Google Play Store u otras tiendas de aplicaciones.
2. **Riesgos de seguridad:** si alguien más tiene acceso a tu clave perdida o archivo JKS, puede firmar y distribuir aplicaciones que parecen ser tuyas, lo que podría dañar tu reputación o robar datos confidenciales de los usuarios.
3. **Pérdida de datos de la aplicación:** si no puede firmar nuevas versiones de su aplicación con la misma clave, los usuarios que hayan instalado la versión anterior no podrán actualizar su aplicación a la nueva versión. Esto puede resultar en la pérdida de datos de la aplicación u otros problemas para los usuarios.

Entre generar el archivo .jks
con línea de comando y con
Android Studio, ¿Cuál te
pareció más fácil y por qué?





Próxima sesión...

- *Diferencias entre apk y bundle*
- *Creación de apk de release*
- *Creación de bundle de release*

{desafío}
latam_

*Academia de
talentos digitales*

