



UA.MASTER MÓVILES

SERVICIOS DE LAS PLATAFORMAS MÓVILES

Servicios de Autenticación



Google Sign-In for Android

Permite que los usuarios accedan a nuestras aplicaciones de forma rápida y segura mediante un sistema de registro que ya conocen, utilizan y en el que confían: su cuenta de Google.

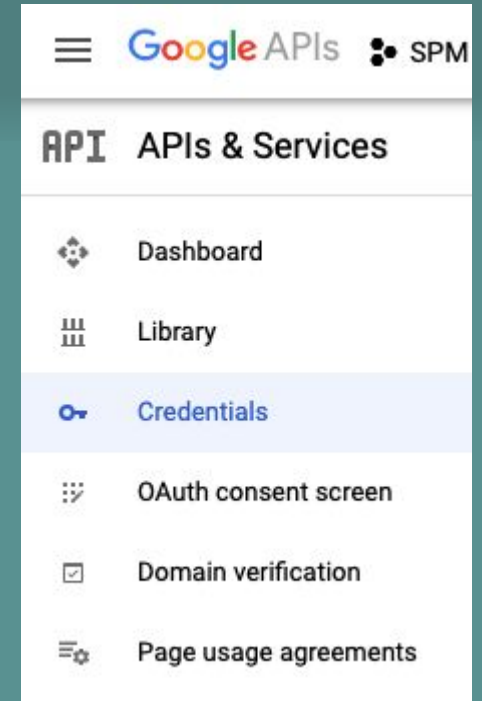
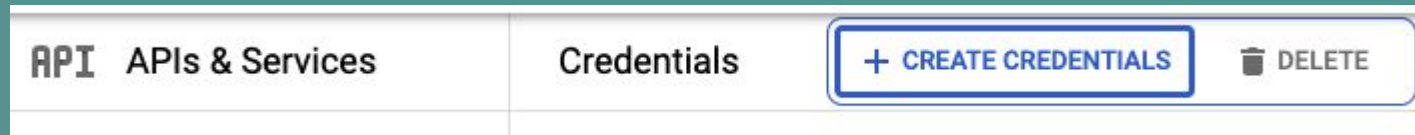
`com.google.android.gms:play-services-auth:xx.xx.xx`

<https://developers.google.com/identity/sign-in/android/>

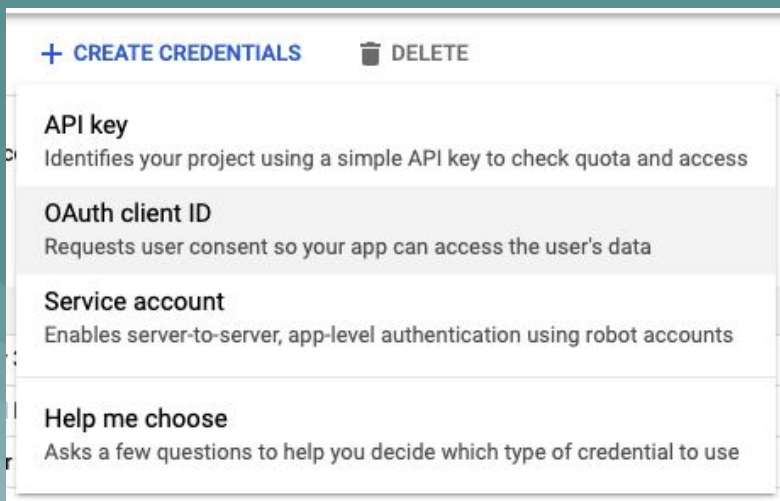
Google Sign-In for Android

Habilitar el servicio Google Sign-In

1. Ir a la dirección <https://console.developers.google.com>.
2. Seleccionar un proyecto existente o crear uno nuevo.
3. Seleccionar **Credentials**.
4. Pulsar el botón para crear unas credenciales nuevas para nuestra aplicación.



5. Seleccionar el tipo de credencial: OAuth Client ID



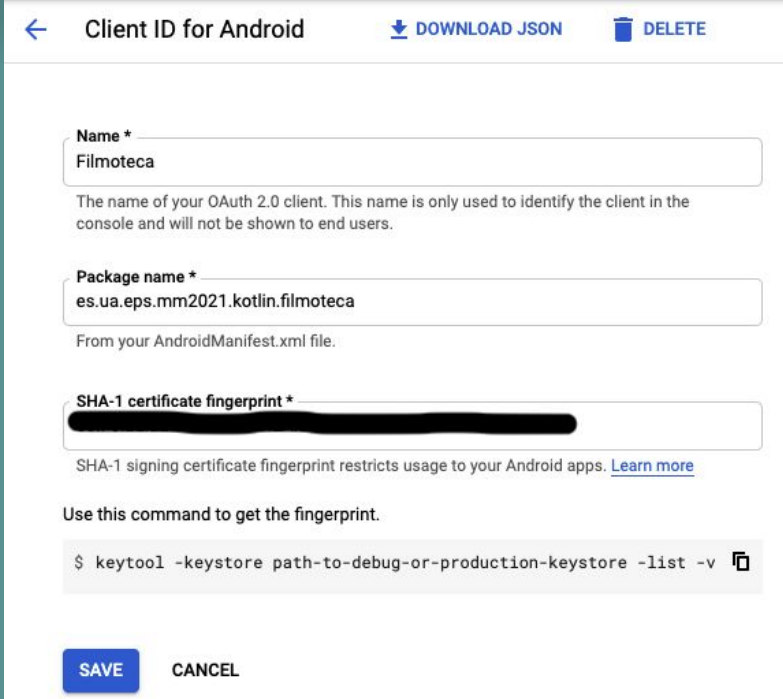
Google Sign-In for Android

Habilitar el servicio Google Sign-In

6. Elegir el tipo de aplicación: Android
7. Añadir el nombre: Filmoteca
8. Añadir el Package Name: es.ua.eps.mm2021.kotlin.filmoteca
9. Añadir vuestro SHA-1:
 - a. Descargar e instalar Java (si no lo tuviésemos).
 - b. Aseguraos de que la variable de entorno JAVA_HOME está establecida.
 - c. Abrir un terminal y ejecutar:

```
keytool -genkey -v -keystore [keystore_name].keystore -alias [alias_name] -keyalg RSA -keysize 2048 -validity 10000
```

keystore_name (debug.keystore) y alias_name (AndroidDebugKey) son los nombres que le queramos dar.
 - d. Introducir una clave cuando os la pregunten (o ninguna).
 - e. ¡No olvidéis la clave!



The screenshot shows the 'Client ID for Android' configuration page in the Google Cloud Console. At the top, there are navigation links: a back arrow, 'Client ID for Android', 'DOWNLOAD JSON', and 'DELETE'. The form contains three main sections: 1. 'Name *' with the value 'Filmoteca' and a description: 'The name of your OAuth 2.0 client. This name is only used to identify the client in the console and will not be shown to end users.' 2. 'Package name *' with the value 'es.ua.eps.mm2021.kotlin.filmoteca' and a note: 'From your AndroidManifest.xml file.' 3. 'SHA-1 certificate fingerprint *' with a redacted fingerprint value and a link to 'Learn more'. Below this, it says 'SHA-1 signing certificate fingerprint restricts usage to your Android apps.' and provides a command to get the fingerprint: '\$ keytool -keystore path-to-debug-or-production-keystore -list -v'. At the bottom, there are 'SAVE' and 'CANCEL' buttons.

Google Sign-In for Android

Habilitar el servicio Google Sign-In

10. Mostrad vuestro SHA-1 almacenado en el keystore:

a. Utilizando keytool:

[←](#) **Client ID for Android** [DOWNLOAD JSON](#) [DELETE](#)

Name *
Filmoteca

The name of your OAuth 2.0 client. This name is only used to identify the client in the console and will not be shown to end users.

Package name *
es.ua.eps.mm2021.kotlin.filmoteca

From your AndroidManifest.xml file.

SHA-1 certificate fingerprint *
[REDACTED]

SHA-1 signing certificate fingerprint restricts usage to your Android apps. [Learn more](#)

Use this command to get the fingerprint.

```
$ keytool -keystore path-to-debug-or-production-keystore -list -v
```

SAVE **CANCEL**

Google Sign-In for Android

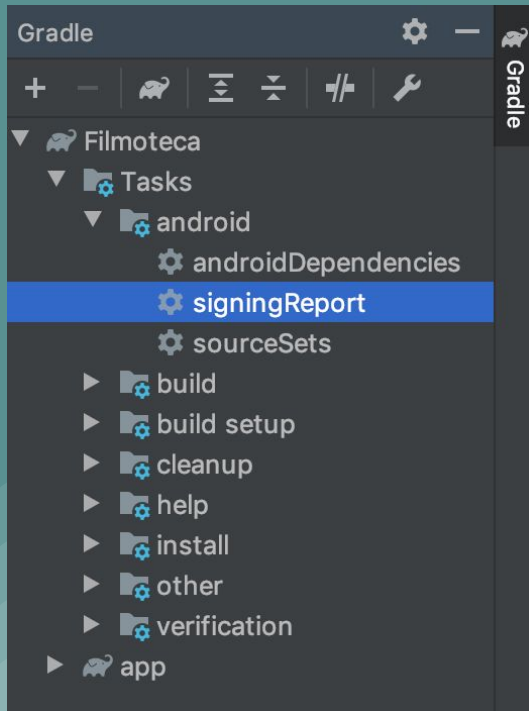
Habilitar el servicio Google Sign-In

10. Mostrad vuestro SHA-1 almacenado en el keystore:

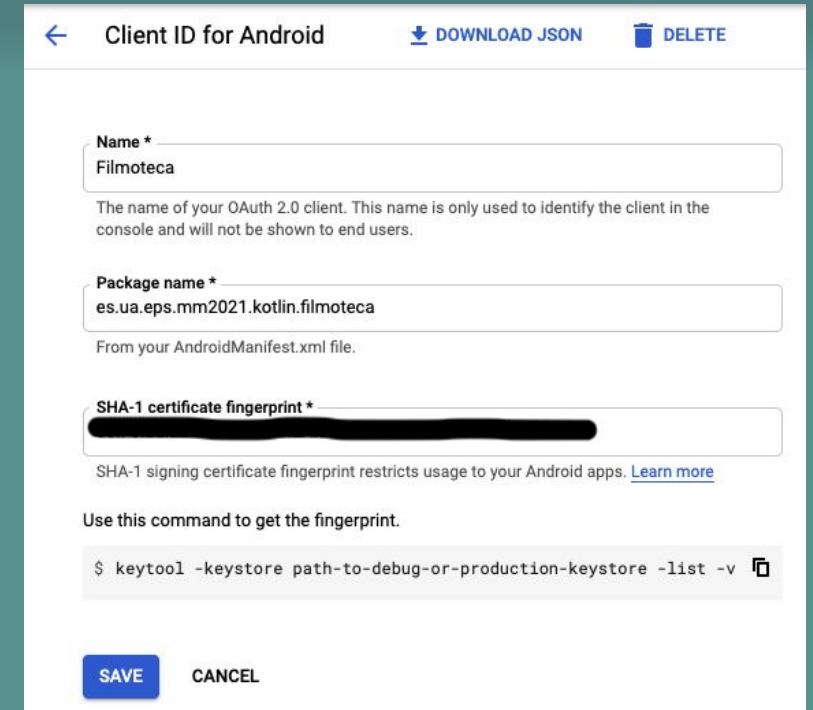
a. Utilizando keytool:

```
keytool -keystore ~/.android/debug.keystore -list -v
```

b. Desde Android Studio podemos mostrar los hashes de nuestros almacenes de claves (si ya existen):



```
> Task :app:signingReport
Variant: debug
Config: debug
Store: /Users/caragones/.android/debug.keystore
Alias: AndroidDebugKey
MD5: [REDACTED]
SHA1: [REDACTED]
SHA-256: [REDACTED]
Valid until: Thursday, October 29, 2048
-----
Variant: release
```



← Client ID for Android [DOWNLOAD JSON](#) [DELETE](#)

Name *
Filmoteca

The name of your OAuth 2.0 client. This name is only used to identify the client in the console and will not be shown to end users.

Package name *
es.ua.eps.mm2021.kotlin.filmoteca

From your AndroidManifest.xml file.

SHA-1 certificate fingerprint *
[REDACTED]

SHA-1 signing certificate fingerprint restricts usage to your Android apps. [Learn more](#)

Use this command to get the fingerprint.

```
$ keytool -keystore path-to-debug-or-production-keystore -list -v
```

[SAVE](#) [CANCEL](#)

Google Sign-In for Android

Habilitar el servicio Google Sign-In

10. Mostrad vuestro SHA-1 almacenado en el keystore:
 - a. Utilizando keytool:

```
keytool -keystore ~/.android/debug.keystore -list -v
```
 - b. Desde Android Studio podemos mostrar los hashes de nuestros almacenes de claves (si ya existen):
 - c. Utilizando gradle wrapper desde la consola:

```
gradlew signingReport
```

[←](#) **Client ID for Android** [DOWNLOAD JSON](#) [DELETE](#)

Name *
Filmoteca

The name of your OAuth 2.0 client. This name is only used to identify the client in the console and will not be shown to end users.

Package name *
es.ua.eps.mm2021.kotlin.filmoteca

From your AndroidManifest.xml file.

SHA-1 certificate fingerprint *
[REDACTED]

SHA-1 signing certificate fingerprint restricts usage to your Android apps. [Learn more](#)

Use this command to get the fingerprint.

```
$ keytool -keystore path-to-debug-or-production-keystore -list -v
```

[SAVE](#) [CANCEL](#)

```
C:\Work\UA\SPM\Filмотeca>gradlew signingReport
```

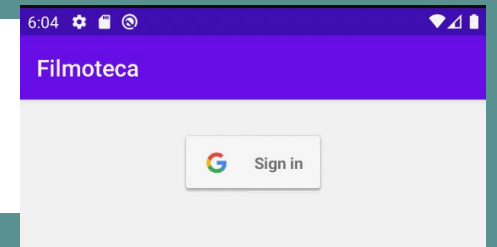
[illegible]

Google Sign-In for Android

Proceso de identificación:

1. Añadir la dependencia al fichero build.gradle de la app.
`com.google.android.gms:play-services-auth:xx.xx.xx`
2. Incluir el botón “Sign in with Google” (Opcional)

```
<com.google.android.gms.common.SignInButton
    android:id="@+id/sign_in_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```



3. Personalizar el estilo del botón (Opcional) y registrar el evento OnClick

```
sign_in_button.setSize(SignInButton.SIZE_STANDARD)
sign_in_button.setOnClickListener(this)
```

4. Configurar las opciones de inicio de sesión (GoogleSignInOptions).

```
override fun onCreate(savedInstanceState: Bundle?) {
    val gso : GoogleSignInOptions = GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
        .requestId()
        .requestProfile()
        .requestEmail()
        .build();
    ...
}
```


Google Sign-In for Android

Proceso de identificación:

4. Crear el objeto GoogleSignInClient.

```
override fun onCreate(savedInstanceState: Bundle?) {  
    UserData.gsClient = GoogleSignIn.getClient(this, gso)  
    ...  
}
```

5. Comprobar si ya hay un usuario identificado.

```
override fun onStart() {  
    val account : GoogleSignInAccount? = GoogleSignIn.getLastSignedInAccount(this)  
    handleAccount(account)  
    ...  
}
```

6. Iniciar el flujo de inicio de sesión.

```
override fun onClick(v: View?) {  
    if (v == sign_in_button) {  
        signIn()  
    }  
}  
  
fun signIn() {  
    val intent : Intent? = UserData.gsClient?.getSignInIntent()  
    startActivityForResult(intent, PLAY_SERVICES_SIGN_IN)  
}
```

Google Sign-In for Android

Proceso de identificación:

6. Iniciar el flujo de inicio de sesión (sigue).

```
override fun onActivityResult (requestCode: Int, resultCode: Int, data: Intent?) {
    super.onActivityResult (requestCode, resultCode, data)
    if(requestCode == PLAY_SERVICES_SIGN_IN) {
        val task : Task<GoogleSignInAccount> = GoogleSignIn. getSignedInAccountFromIntent (data)
        handleSignInResult (task)
    }
}

fun handleSignInResult (completedTask: Task<GoogleSignInAccount>) {
    try {
        val account : GoogleSignInAccount ? = completedTask. getResult (ApiException:: class .java)
        handleAccount (account)
    }
    catch (e: ApiException) {
        Log.d("GPS", "SignInResult Failed: ${e.statusCode} ")
    }
}

private fun handleAccount (account: GoogleSignInAccount ?) {
    if(account != null) {
        UserData.account = account
        // Initiate our real main activity
    }
}
```

Google Sign-In for Android

Proceso de identificación:

7. Obtener información del perfil.

```
if(account != null) {  
    Log.d("GPS", "id: ${account.id}")  
    Log.d("GPS", "idToken: ${account.idToken}")  
    Log.d("GPS", "serverAuthCode: ${account.serverAuthCode}")  
    Log.d("GPS", "givenName: ${account.givenName}")  
    Log.d("GPS", "familyName: ${account.familyName}")  
    Log.d("GPS", "displayName: ${account.displayName}")  
    Log.d("GPS", "email: ${account.email}")  
    Log.d("GPS", "photoUrl: ${account.photoUrl}")  
    ...  
}
```

8. Finalizar la sesión (recomendable).

```
private fun signOut() {  
    UserData.gsClient?.signOut()?.addOnCompleteListener(this) { ... }  
}
```

9. Desconexión de la cuenta (altamente recomendable).

```
private fun disconnect() {  
    UserData.gsClient?.revokeAccess()?.addOnCompleteListener(this) { ... }  
}
```