Integración SDK SIGNply

Android

versión 0.5.3

1. Introducción

Bienvenid@ a la documentación de la integración del **SDK SIGNply**. Por medio de este SDK podrá añadir la funcionalidad de firmar un fichero **PDF** sin apenas esfuerzo. En el presente documento se incluye el manual de integración del SDK para perfiles técnicos.

2. Contenido

- **SDK:** Se facilita el SDK por medio del fichero SignplySDK.aar compatible con dispositivos físicos y con simuladores con arquitectura de 64 bits (x86_64). También con algunos simuladores con arquitectura 32 bits (x86) pero solamente con los sistemas Android 9 y 11.

  https://developer.android.com/studio/releases/emulator#30-0-0.

- **Licencia:** Se facilita un archivo de licencia para poder utilizar el SDK.
- **Manual de integración**: Hace referencia a este documento.
- **Proyecto de ejemplo completo Kotlin**: Ejemplo completo escrito en Kotlin de llamada al *SDK* de *SIGNply* para poder facilitar la integración. Puede servir cómo un punto de entrada para el proyecto a desarrollar. Contiene una licencia demo.
- **Proyecto de ejemplo simple Java:** Ejemplo simple escrito en Java de llamada al *SDK* de *SIGNply* para poder facilitar la integración. Puede servir cómo un punto de entrada para el proyecto a desarrollar. Contiene una licencia demo.

3. Requisitos previos

Este *SDK* es compatible desde la versión de *Android 5 (API 21)* tanto con dispositivos físicos (smartphones y tablets), simuladores de 64 bits y algunos simuladores modernos de 32 bits (Android 9 o Android 11).. Es necesario establecer la *versión mínima **21*** en el build.gradle módulo app**.** Compila con la versión del SDK **30**. La versión de **Android Studio** utilizada es la **4.1.**

Para una correcta integración es necesario añadir abiFilters “arm64-v8a”, “armeabi-v7a” , “x86_64”. Se muestra en la imagen inferior.

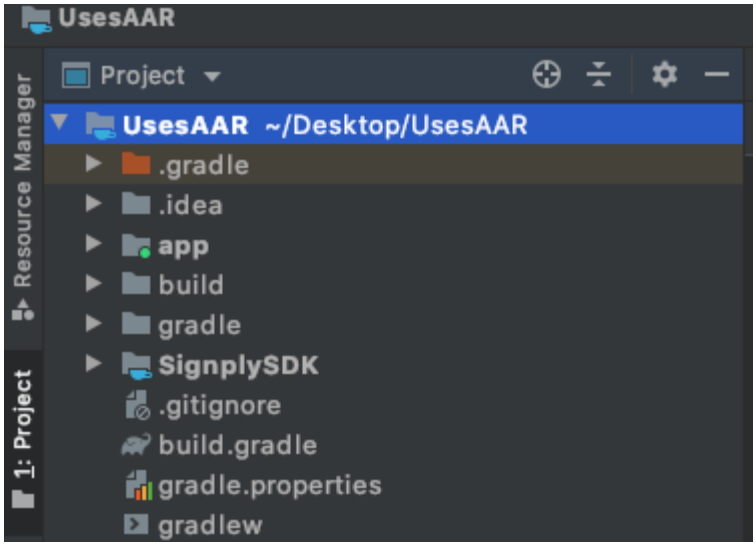
```
android {
    compileSdkVersion 30
    buildToolsVersion "29.0.3"

    defaultConfig {
        applicationId [redacted]
        minSdkVersion 21
        targetSdkVersion 30
        versionCode 2
        versionName "1.0.3"
        ndk {
            abiFilters "arm64-v8a", "armeabi-v7a", "x86_64"
        }
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }
```

Añadir el SDK es muy sencillo desde Android Studio:

File -> New Module -> Import .JAR / .AAR Package -> (seleccionar SignplySDK_*versionNumber.aar).*

A continuación para asegurarse de que el SDK se ha añadido correctamente es necesario comprobar que la carpeta SignplySDK se encuentra dentro de los archivos en la pestaña Project.



Dentro de las dependencias del proyecto build.gradle (módulo app) es necesario añadir las siguientes líneas y sincronizar el proyecto.

```
implementation 'androidx.appcompat:appcompat:1.3.0' //SDK
implementation 'androidx.core:core-ktx:1.5.0' //SDK
implementation 'com.google.android.material:material:1.3.0' //SDK
implementation 'androidx.constraintlayout:constraintlayout:2.0.4' //SDK
implementation 'com.google.android.gms:play-services-location:18.0.0' //SDK
implementation project(':SignplySDK_*version_code*')
```

4. Utilización

El SDK de SIGNply es lanzado a través de un Intent, a este Intent es necesario pasarle un objeto de tipo *SignplySDKParams*.

```
val intent = Intent(this, SignplySDKLauncher::class.java)
intent.putExtra("signplySDKParams", signplySDKParams)
startActivityForResult(intent, REQUEST_CODE_LAUNCH_SIGNPLY_SDK)
```

El resultado se obtiene en el onActivityResult. En el caso de que el documento se haya firmado correctamente devuelve una **Uri** con el documento firmado.

En caso contrario, devuelve un extra del Intent con la clave *error* y el error correspondiente como valor.

4.1 Parámetros principales de llamada

SIGNply dispone de una serie de parámetros de llamada para la configuración del firmado de documento que se exponen a continuación:

signplySDKParams (SignplySDKParams): estructura de datos que incluye toda la configuración necesaria.

4.2 Parámetros de SIGNply

Para facilitar la integración, la mayoría de parámetros son opcionales. El objeto SingplySDKParams se puede formar tan solo con esta mínima configuración en Kotlin.

```
val signplySDKParams = SignplySDKParams(license, selectedFileUri)
```

Licencia de tipo *String*, y selectectedFileUri de tipo *Uri*.

A continuación se muestran todos los parámetros disponibles y configurables.

SignplySDKParams :

- **licenseB64** (String) : licencia en base 64 de ESS
- **uri** (Uri): Uri del documento pdf.

  **El uri prevalece sobre el documentB64.*
- **documentB64** (String): string en base64 del documento pdf.
- **saveDocumentSignedName** (String): nombre de documento firmado. Si no se especifica, por defecto obtiene el valor "documento_firmado.pdf" y lo guarda en la carpeta propia de la aplicación. Si el nombre ya existe se concatena un (1), (2), (3) y así sucesivamente
- **isPrivate** (Boolean): Indica si el documento se debe guardar de manera privada para que no puedan acceder otras apps al documento firmado *getFilesDir()* o en caso contrario para ser accesibles desde otras apps *getExternalFilesDir()*.

  **Los documentos se eliminan cuando se desinstala la aplicación. Ver más en:*

  https://developer.android.com/training/data-storage
- **widget** (SignplySDKWidget): widget de firma configurable
- **tsp** (SignplySDKTSP): TSP configurable

- **header** (SignplySDKHeader): parámetros de la cabecera de la firma
- **extra** (SignplySDKExtra): funcionalidad extra en el SDK configurable
- **certificate** (SignplySDKCertificate): certificado de firma configurable
- **crypto** (SignplySDKCripto): parámetros para desencriptar el documento

SignplySDKWidget

- **widgetType** (SignplySDKWidgetType): tipo de posicionamiento del widget. Por defecto usa el tipo manual.
- **widgetFloatText** (String): indica la cadena a buscar dentro del documento dónde se incrusta el widget. Devuelve el primer resultado obtenido. Este parámetro es sensible a mayúsculas y a minúsculas.
- **widgetFieldFieldname** (String): nombre de campo preexistente en el que se incrusta el widget.
- **widgetManualRatio** (Float): relación entre el ancho y el alto de widget para el posicionamiento manual. Por defecto 2,5. El valor debe estar entre 1 y 4. Si es menor que 1, toma valor 1 y si es mayor de 4 toma valor 4.
- **widgetFixedPage** (Int): número de página en el que se incrusta el widget. Empieza en 1.

  Si el valor es 0 firma en todas las páginas.

  Si el valor es mayor que el número total de páginas se firma en la última página.
- **widgetFixedX** (Int): indica el desplazamiento en horizontal de la posición del widget desde el vértice inferior izquierdo de la página de un documento.
- **widgetFixedY** (Int): indica el desplazamiento en vertical de la posición del widget desde el vértice inferior izquierdo de la página de un documento.
- **widgetFloatGapY** (Int): indica el número entero de unidades de desfase (positivo o negativo) en vertical desde la parte inferior al texto
- **widgetFloatGapX** (Int): indica el número entero de unidades de desfase (positivo o negativo) en horizontal desde la parte inferior al texto

- **widgetCustomText** [SignplySDKWidgetCustomText]: Array multilínea con los textos a incrustar en el widget de firma
- **widgetWidth** (Int): ancho del widget. Valor mínimo 50. El valor por defecto es 150.
- **widgetHeight** (Int): alto del widget. Valor mínimo 50. El valor por defecto 75.

SignplySDKTSP

- **tspActivate** (Bool): activación de introducción de sello de tiempo
- **tspURL** (String): url de TSP
- **tspUser** (String): usuario de TSP
- **tspPassword** (String): password de TSP

SignplySDKHeader

- **author** (String): Autor de la firma
- **reason** (String): Razón de la firma
- **contact** (String): Contacto del autor o entidad de la firma
- **location** (String): Localización del autor o entidad de la firma

SignplySDKExtra

- **autoOpen** (Boolean): Permite desplegar el widget de firma automáticamente al abrir el documento. (viewLastPage debe ser false).
- **captureSignatureSeconds** (Int): Tiempo previo a que se cierre el widget de firma
- **viewLastPage** (Bool): Si es true, es necesario que el usuario visualice hasta la última página para poder firmar
- **showReject** (Bool): Si es true muestra un botón de rechazar documento
- **showShare** (Bool): Si es true muestra un botón de compartir documento

- **displayName** (String): Nombre visible del PDF en el título de su visualización

SignplySDKCertificate

- **signCertP12B64** (String): Certificado de firma en base64
- **signCertPassword** (String): Contraseña del certificado de firma
- **encKeyB64** (String): Clave pública de cifrado de los datos biométricos en base64

SignplySDKWidgetType

- **manual:** Posicionamiento modo manual en el cual el usuario puede mover la firma libremente
- **field:** Posicionamiento que busca un campo de firma en concreto
- **fixed:** Posicionamiento fijo en el documento, se le pasa un número de página y una posición relativa dentro del documento
- **float:** Posicionamiento flotante en el documento en base a una búsqueda de cadena de texto

SignplySDKWidgetCustomText

- **fontSize** (Int): Tamaño del texto
- **text** (String): Cadena de texto

SignplySDKCrypto

- **isEncrypted** (Bool): Indica si el documento pasado por Uri está encriptado
- **key** (String): Clave AES para desencriptar.
- **fileNameOnCacheDir** (String): Nombre del archivo en el directorio caché a desencriptar.

4.3 Resultados de SIGNply

El resultado se obtiene en el onActivityResult. En el caso de que el documento se haya firmado correctamente devuelve una *Uri* con el documento firmado.

En caso de que que el documento se haya rechazado devuelve un código de resultado -2 y el motivo del rechazo en el extra del Intent con la clave *reject_reason*

En caso de error, devuelve un extra del Intent con la clave *error* y el error correspondiente como valor.

https://developer.android.com/training/basics/intents/result?hl=es-419

4.4 Anotaciones

Los documentos que se hayan guardado y que no sean privados se pueden visualizar desde la carpeta de aplicación. Para ello es necesario acceder a la siguiente carpeta: Dentro del Almacenamiento es necesario ir a Android -> data -> *(identificador de la aplicación)* -> files

En esta carpeta se encuentran todos los ficheros PDF que hayan sido firmados desde el SDK de SIGNply.

5. Permisos

El SDK no declara permisos explícitos en el manifiesto (*AndroidManifest.xml*).
Para que funcione el TSP (Sellado de Tiempo) correctamente es necesario añadir el siguiente permiso desde el archivo de manifiesto de la aplicación dado que se necesita acceso a Internet:

```
<uses-permission android:name="android.permission.INTERNET" />
```

Si es necesario recoger la ubicación del usuario en el momento de la firma será necesario añadir el siguiente permiso en el manifiesto de la aplicación:

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

En el caso de que se declare, si el usuario no ha concedido todavía el permiso de ubicación, se le requerirá en tiempo de ejecución cuando abra la caja de firma.

** En algunos simuladores es necesario añadir el permiso *ACCESS_FINE_LOCATION* para una correcta ejecución.

6. Estilos

El SDK usa los recursos de colores preestablecidos en el ecosistema Android. Estos pueden ser sobreescritos desde la aplicación. Desde styles.xml y colors.xml se pueden establecer los colores principales de un tema: *colorPrimary*, *colorPrimaryDark*, *colorAccent*

https://developer.android.com/guide/topics/ui/look-and-feel/themes