Scangine SDK
(Android)

1. Add camera permissions to your manifest file:

Add these lines:

```
<uses-permission android:name="android.permission.CAMERA" />
<uses-feature android:name="android.hardware.camera" />
<uses-feature android:name="android.hardware.camera.autofocus" />
```

2. Include Scangine.aar library to your project

Click File > New > New Module.

Click Import .JAR/.AAR Package then click Next.

Enter the location of the Scangine.aar file then click Finish.

Make sure the library is listed at the top of your settings.gradle file, as shown here

```
include ':app', 'scangine'
```

Open the app module's build.gradle file and add a new line to the dependencies block as shown in the following snippet:

```
dependencies {
.....
compile project(":scangine")

}
```

Click Sync Project with Gradle Files.

3. Implement

```
public interface ScangineSDKListener {

void onScanResult(String barcode, String barcodeType);

String getAction1Text();

String getAction2Text();

void onAction1();

void onAction2();
}
```

There are 3 action buttons at the bottom of the scanner. First one is "Close" and closes the scanner. The other 2 can be set and used by the ScangineSDKListener by these methods. Use the first 2 methods to define the text of the buttons.
These methods are called once before the scanner starts. The onAction1() and onAction2() callbacks are called on button pressed by the user.

You need to have and keep an instance of the class ScangineSDK to use the SDK or to make your activity to extend CameraActivity.
In first case make a member of type ScangineSDK in your activity- for example _scangineSDK.
You have to call

```
_scangineSDK.requestUserCameraPermission(myActivity);
```

in your activity's void onCreate(Bundle savedInstanceState)
to request the camera permission.

Now you are ready to start the scan dialog in your app at any time.
To do so just call:

```
_scangineSDK.setScanTitle(dialogTitle);
_scangineSDK.startScan(myActivity,myScangineSDKListener);
```

on every scan result your ScangineSDKListener's

```
void onScanResult(String barcode, String barcodeType);
```

will be called with the result barcode and barcodeType.

When you handle the code - if you want to display some information on the Scandialog - you can call:

```
_scangineSDK.setProductInfo(text_line1,text_line2);
```

To programatically stop scan - call:

```
_scangineSDK. stopScan();
```

As a option you can enable or disable barcode type decoders by calling thse methods:

```
_scangineSDK. setON_EAN13_UPCA(true_or_false);

_scangineSDK.setON_EAN8(true_or_false);

_scangineSDK. setON_UPCE(true_or_false);
```