Slyce Android SDK

Version 2.1

Last updated: July 13th, 2015

Contents

GETTING STARTED

Overview

The Slyce Android SDK enables Android developers to easily interact with the Slyce image recognition platform.
The SDK provides the methods required to submit images and receive results. It doesn't contain any UI as of the current version.

Prerequisites

- Minimum Android OS versions 4.0 (API level 14) and higher.
- Android Studio development environment
- A Slyce client ID

Setup

1.Create libs folder and place the Slyce AAR file inside.

2.Add at build.grade:

```
repositories{
 flatDir {
      dirs 'libs'
       }
      }
```

3.Add at build.grade dependencies

```
compile(name:'slyce', ext:'aar')
```

CODE INTEGRATION

SlyceRequest

```
// Implement OnSlyceRequestListener:
public class MainActivity extends Activity implements OnSlyceRequestListener, {

    @Override
    public void onSlyceProgress(final long progress, final String message, String token) {
            // progress - progress percentage
            // message - progress message
            // requestToken - request unique id}

    @Override
    public void onImageDetected(String productInfo) {
            // productInfo - representing a short info about the matched 2D products}

    @Override
    public void onImageInfoReceived(JSONArray products) {
            // products - representing the additional info}

    @Override
    public void onBarcodeDetected(SlyceBarcode barcode) {}

    @Override
    public void onResultsReceived(final JSONObject products) {
      // products - founds products (might be empty if no products found)
    }

    @Override
    public void onError(final String message) {
            // message - error message }

    @Override
    public void onSlyceRequestStage(SlyceRequestStage message) {
            // message - of type StageMessage (enum) indicates stage has been completed.
            // For example: this call back will be invoked after a bitmap has been uploaded to the}

    @Override
    public void onItemDescriptionReceived(JSONObject keywords) {
            // keywords - items's keywords description }

    @Override
    public void onFinished() {}
}
…
}
```

```
// Create Slyce singleton object:
slyce = Slyce.getInstance(this);

// Initiate Slyce SDK with OnSlyceOpenListener

slyce.open("YOUR CLIENT ID", new OnSlyceOpenListener() {

        @Override
        public void onOpenSuccess() {}

        @Override
        public void onOpenFail(String message) {}
  });


// Create SlyceRequest object for searching products by image or by image url

SlyceRequest request = new SlyceRequest(slyce, this, new JSONObject());

// Searching products by image url
String imageUrl = "http://…";
request.getProducts(imageUrl);

// Searching products by image (Bitmap)
Bitmap bitmap;
request.getProducts(bitmap);

// Cancelling SlyceRequest
request.cancel();
```

// SlyceCamera:

*** Scanning products/barcodes/QR codes.**
*** Managing the camera and displaying its preview**

Create a CameraActivity and Implement OnSlyceCameraListener:

```
public class CameraActivity extends Activity implements OnSlyceCameraListener {

   @Override
   public void onCameraBarcodeDetected(SlyceBarcode barcode) {
       // Called when barcode is found
   }

   @Override
   public void onCameraImageDetected(String productInfo) {
       // Called when 2D products are found
   }

   @Override
   public void onCameraImageInfoReceived(JSONArray products) {
      // Called when additional info for the previously recognized 2D product is found.
   }

   @Override
   public void onCameraSlyceProgress(long progress, String message, String token) {
     //  Reporting a numeric value and informative message.
   }
```

```
    @Override
    public void onCameraSlyceRequestStage(SlyceRequestStage message) {
        // Reporting the stage currently being processed.
    }

    @Override
    public void onCameraResultsReceived(JSONObject products) {
        // Called when 3D products are found
    }

    @Override
    public void onSlyceCameraError(String message) {
        // Called when an error occured
    }

    @Override
    public void onTap(float x, float y) {
        // Called when the camera was touched in a specific point.
    }

    @Override
    public void onSnap(Bitmap bitmap) {
        // Called when the snapped bitmap is ready after SlyceCamera.snap() was invoked
    }

    @Override
    public void onCameraFinished() {
       // Called when Slyce search process ended
    }
```

// Create and initiate Slyce single object as mentioned earlier

// UI:
The SlyceCamera constructor expects an empty SurfaceView, it will take care of displaying the camera preview.
SurfaceView should be added to the Activity xml file.

Create activity_camera.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">

    <SurfaceView
        android:id="@+id/preview"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</RelativeLayout>
```

Please add android:configChanges to the CameraActivity at your manifest.xml

```
        <activity
            android:name=".CameraActivity"
            android:configChanges="orientation|screenSize"
         </activity>
```

Now you can create the SlyceCamera object, it requires:
* Parent Activity
* Opened Slyce object
* The surface
* Options - optional

* OnSlyceCameraListener for notifying results

```
// Create the SlyceCamera object
slyceCamera = new SlyceCamera(this, Slyce.getInstance(this), preview, null, this);
```

You need to handle the life cycle of SlyceCamera:

```
@Override
protected void onResume() {
    super.onResume();
    slyceCamera.start();
}

@Override
protected void onPause() {
    super.onPause();
    slyceCamera.stop();
}

@Override
protected void onDestroy() {
    super.onDestroy();
    if(slyce != null){
        slyce.close();
    }
}
```

Now you can start scanning images/barcodes

SlyceCameraFragment

Full UI implementation of SlyceCamera.

Create FullUIActivity and its xml file activity_full_ui.xml

Please add android:configChanges to the CameraActivity at your manifest.xml

```
<activity
  …
  android:configChanges="orientation|screenSize"
  …
</activity>
```

Please add a container for SlyceCameraFragment at activity_full_ui.xml

```
<FrameLayout
    android:id="@+id/slyce_camera_fragment_container"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
</FrameLayout>
```

Adding SlyceCameraFragment after Slyce SDK successfully opened.

```
Slyce slyce = Slyce.getInstance(activity);

slyce.open(clientID, new OnSlyceOpenListener() {
    @Override
    public void onOpenSuccess() {
     // Add SlyceCameraFragment to the FullUIActivity
    }

    @Override
    public void onOpenFail(String message) {
```

```
        }
    });
```

Add SlyceCameraFragment to the FullUIActivity

SlyceCameraFragment.newInstance() expects 3 parameters
1. JsonObject Options - optional (can be null)
2. boolean - enabling/disabling the scanner
3. boolean - enabling/disabling an automatic resume for scanner after image/barcode detection

```
SlyceCameraFragment slyceFragment = SlyceCameraFragment.newInstance(null, true, true);
FragmentTransaction transaction = getFragmentManager().beginTransaction();
    transaction.replace(R.id.slyce_fragment_container, slyceFragment);
    transaction.addToBackStack(null);
    transaction.commit();
```

Its important to add SlyceCameraFragment to your Activity BackStack

Add onDestroy method

```
@Override
protected void onDestroy() {
    super.onDestroy();
    if(slyce != null){
        slyce.close();
    }
}
```

OnSlyceCameraFragmentListener

In order to receive results please implement OnSlyceCameraFragmentListener at your Activity. Please note its a must!

```
public class SlyceActivity extends Activity implements OnSlyceCameraFragmentListener {

    @Override
    public void onCameraFragmentBarcodeDetected(SlyceBarcode slyceBarcode) {
      // Called when barcode is found
    }

    @Override
    public void onCameraFragmentImageDetected(String info) {
      // Called when 2D products are found
    }

    @Override
    public void onCameraFragmentImageInfoReceived(JSONArray products) {
      // Called when additional info for the previously recognised 2D product is found.
    }

    @Override
    public void onCameraFragmentResultsReceived(JSONObject results) {
        // Called when 3D products are found
    }

    @Override
    public void onCameraFragmentError(String message) {
        // Called when an error occured
    }

    @Override
    public void onCameraFragmentFinished() {
        // Called when Slyce search process ended
    }
 }
…
}
```

- execute can be called only once per SlyceProductsRequest
- please note that any call to execute should be triggered after Slyce SDK was successfully opened (initialised).
- requestToken is a unique identifier per a request
- canceled request cannot be resumed

Migrating from SDK 1.x to 2.x

SlyceRequest:
SlyceProductsRequest changed to **SlyceRequest** and it has only one constructor now.

OnSlyceRequestListener methods

* Changed:
  * on2DRecognition changed to onImageDetected
  * on3DRecognition changed to onResultsReceived
  * onStageLevelFinish changed to onSlyceRequestStage
* Added:
  * onBarcodeDetected
  * onImageInfoReceived
  * onFinished
  * onItemDescriptionReceived

New Methods:
getProducts(Bitmap), getProducts(String) for getting a list of products.

getItemDescription(Bitmap), getItemDescription(String) for getting a keywords description of the given image (bitmap/url)

Example code: getting products with image url

```
SlyceRequest request = new SlyceRequest(slyce, this, new JSONObject());
request.getProducts(image_url);
```

Permissions - no need to add app permissions at the AndroidManifest.xml

Slyce singletone:

Slyce.getInstance(Context context) takes only one parameter now.

“ClientID” should be passed now to Slyce.open(…) method.

Example:

```
Slyce slyce = Slyce.getInstance(this);

slyce.open(clientId, new OnSlyceOpenListener() {

    @Override
    public void onOpenSuccess() {

    }

    @Override
    public void onOpenFail(String message) {

    }
});
```