Integration Steps for Native Android Application with Netcore SDK for FCM

(https://github.com/NetcoreSolutions/Smartech-Android-FCM-SDK)

Prerequisites:-

1: Get **google-services.json** file and **FCM Server Key** from **Firebase Developer console**

2: Get **App Id** and **Smartech FCM SDK** from **Smartech Panel**

For FCM Setup in Application: (Avoid if already setup)

1. **Add google-services.json file in app directory of project**
2. **Add below code in dependencies of project build.gradle file**

   classpath **'com.google.gms:google-services:3.1.0'**
   classpath **'com.android.tools.build:gradle:3.1.3'**

3. **Add below code at last line in app build.gradle file**

   apply plugin: **'com.google.gms.google-services'**

For Integrate SDK:

(supported android minsdkversion 16 or more (i.e. jelly bean and above))

1. **Add below code in dependencies of app build.gradle file** (not in child dependencies)

   implementation 'in.netcore.smartechfcm:smartech-fcm:1.1.3'
   implementation 'com.google.firebase:firebase-messaging:11.6.2'
   implementation 'com.google.code.gson:gson:2.8.0'
   implementation 'com.firebase:firebase-jobdispatcher:0.8.5'

For Push Notification as well as inbuilt activities:

1. **Paste below code in launcher Activity in method onCreate** (above the super.onCreate.line)

   NetcoreSDK.*register*(getApplication(),"<App ID>", "<identity>");

Note: Identity must be **""(blank)** or **Primary key** which defined **on smartech Panel**

For Login Activity add below code for login activity to make user Identified:

NetcoreSDK.*login*(context, "<*identity*>");

For Logout Activity add below code for logout activity to make user anonymous:

```
NetcoreSDK.logout( context, "<identity>" );
```

For custom Activity tracking add below code for Custom activity:

```
NetcoreSDK.track( context, "<identity>",<eventId>,payload);
```

Note: Activity tracking code get from Smartech panel for each custom activity

For Profile Update, Follow Below code

```
JSONObject newProfile = new JSONObject();
try {
  newProfile.put( "NAME", "Developer" );
  newProfile.put( "AGE", 25 );
  newProfile.put( "MOBILE", "1234567890" );
  NetcoreSDK.profile(this, "<identity>", newprofile);
}
catch ( JSONException e ) {
  e.printStackTrace();
}
```

Note: attribute name must be in **Capital**.

To set custom push notification icon add below code in Launching Activity:

```
NetcoreSDK.setPushIcon( context, "<path to drawable icon>");
```

```
e.g. NetcoreSDK.setPushIcon(getApplicationContext(), R.mipmap.ic_launcher)
```

Note: The notification icon should be strictly in **.png format** as per **Google's guidelines** & **Preferable size for push notification icons** is mentioned below:

```
drawable-mdpi :- 24 x 24
drawable-hdpi :- 36 x 36
drawable-xhdpi :- 48 x 48
drawable-xxhdpi :- 72 x 72
drawable-xxxhdpi :- 96 x 96
```

To fetch delivered push notifications add below code in activity:

```
JSONArray jsonNotification = NetcoreSDK.getNotifications(context);
```

Note: The method will return a JSONArray of push notifications delievered

To use push notifications along with custom Firebase Messaging Class add below code in Firebase Messaging Class:

boolean pushFromSmartech = NetcoreSDK.handleNotification(context, remoteMessage);

Note: The behaviour of the method mentioned above is as follows:

- It returns **true**, if the push notification is received from the Smartech panel. In this case, the SDK will do the needful itself without any extra efforts.

- It returns **false**, in case if the push notification is not received from the Smartech panel. In this case, one can handle the push notification as per their custom implementation of Firebase Messaging Class or pass on the instructions to some other SDK as per the choice.

If user wants to opt out from being tracked add below code:

NetcoreSDK.optOut(context, <boolean_flag>);

Note: The method mentioned above accepts a compulsory boolean value (true/false).

- If an end user wants to opt out, the flag should be passed as **true**. Once the user opts out, Netcore SDK will not be able to track that particular user further and no communications will be received by that user.
e.g. **NetcoreSDK.optOut(context, true);**

- If an end user wants to opt in, the flag should be passed as **false**. Once the user opts in, Netcore SDK will be able to track that particular user further and next communications will be received by that user.
e.g. **NetcoreSDK.optOut(context, false);**

To track location of the user add below code:

NetcoreSDK.setUserLocation(context, <double_latitude>, <double_longitude>);

Note: The method mentioned above accepts 3 parameters including context, latitude & longitude. Data type of 'latitude' & 'longitude' should compulsorily be **Double**.
e.g. **NetcoreSDK.setUserLocation(context, 18.9431, 72.8272);**

- In case if any one of the parameters is **null**, the SDK will not be able to persist user location.