

Facebook & Google Sign-In for Android

29 DECEMBRE 2019

Programmation Android



Table des matières

I.	Intégration de Google Sign-In dans votre application Android	3
1.	Ajouter les services de Google Play.....	3
2.	Configurer un projet Google API Console	4
2.1.	Création d'un projet google api	4
2.2.	Intégration d'authentification via Google dans Notre Application Android ..	7
II.	Ajouter Facebook Login à l'application Android	11
1.	Créer une nouvelle application dans la console développeur de Facebook	11
1.1.	Générer votre ID d'application	11
1.2.	Configurer Facebook login.....	12
1.3.	Spécifier le type de votre application	12
1.4.	Ajouter les dépendances	13
1.5.	Ajouter Le package de votre application	13
1.6.	Ajouter vos clés de hachage de développement et de publication	14
1.7.	Activer l'authentification unique pour votre app	15
2.	Configurer notre application Android	16
2.1.	Modifier vos ressources et votre manifeste	16
2.2.	Ajouter le bouton Facebook Login.....	17
2.3.	Enregistrer un rappel	18
2.4.	Vérifier l'état de la connexion	20
III.	Démonstration	22
IV.	Conclusion	22

I. Intégration de Google Sign-In dans votre application Android

Avant de pouvoir commencer à intégrer Google Sign-In dans votre propre application, vous devez configurer un projet Google API Console et configurer votre projet Android Studio. Les étapes de cette page font exactement cela. Les étapes suivantes décrivent ensuite comment intégrer la connexion Google dans votre application.

1. Ajouter les services de Google Play

Dans le fichier `build.gradle` de niveau supérieur de votre projet, assurez-vous que le référentiel Maven de Google est inclus:

```
allprojects {
    repositories {
        google()

        // If you're using a version of Gradle lower than 4.1, you must instead use:
        // maven {
        //     url 'https://maven.google.com'
        // }
    }
}
```

Ensuite, dans votre fichier `build.gradle` au niveau de l'application, déclarez les services Google Play en tant que dépendance:

```
apply plugin: 'com.android.application'
...

dependencies {
    implementation 'com.google.android.gms:play-services-auth:17.0.0'
}
```

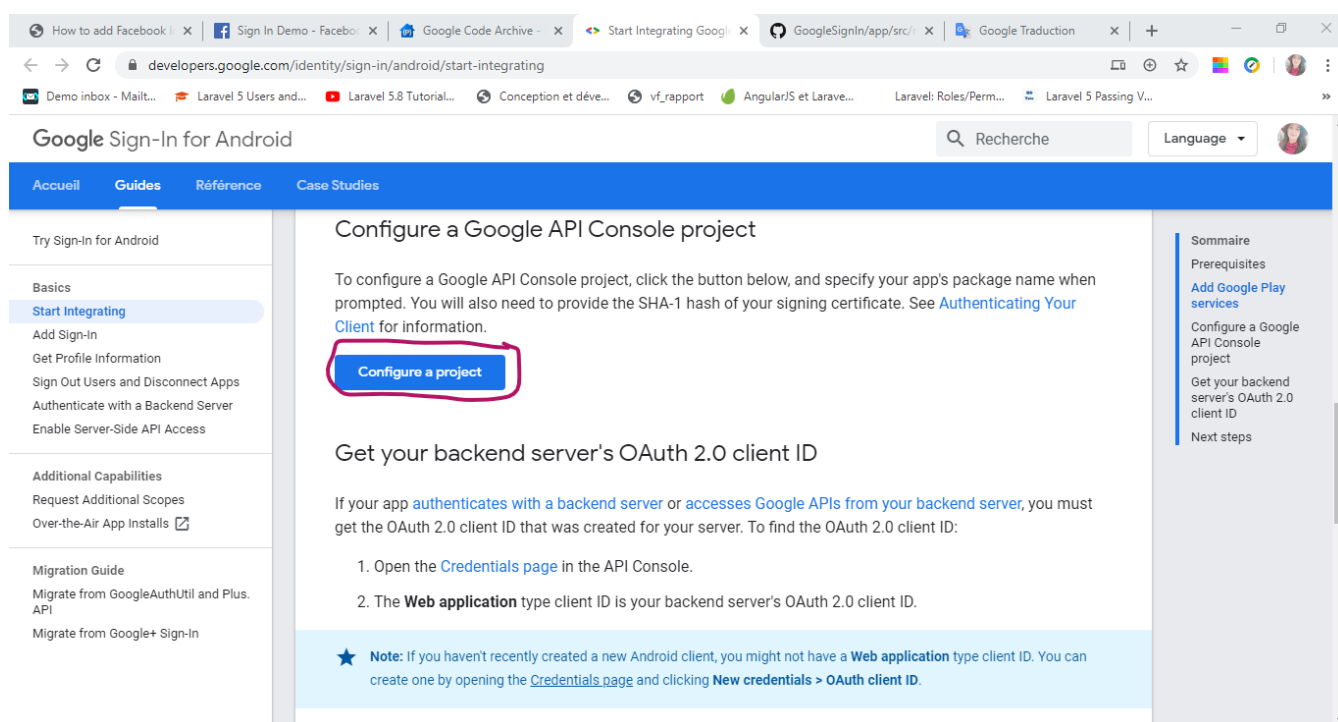
2. Configurer un projet Google API Console

Pour configurer un projet Google API Console, cliquez sur le bouton ci-dessous et spécifiez le nom du package de votre application lorsque vous y êtes invité. Vous devrez également fournir le hachage SHA-1 de votre certificat de signature.

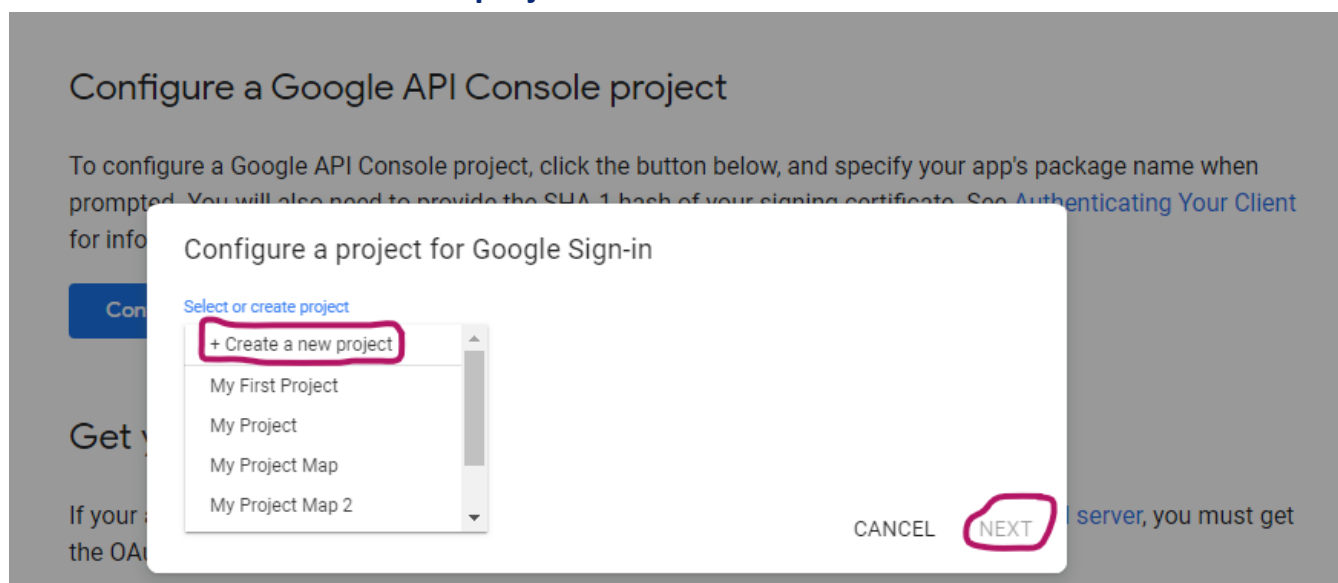
2.1. Création d'un projet google api

Premièrement vous devez aller sur ce site :

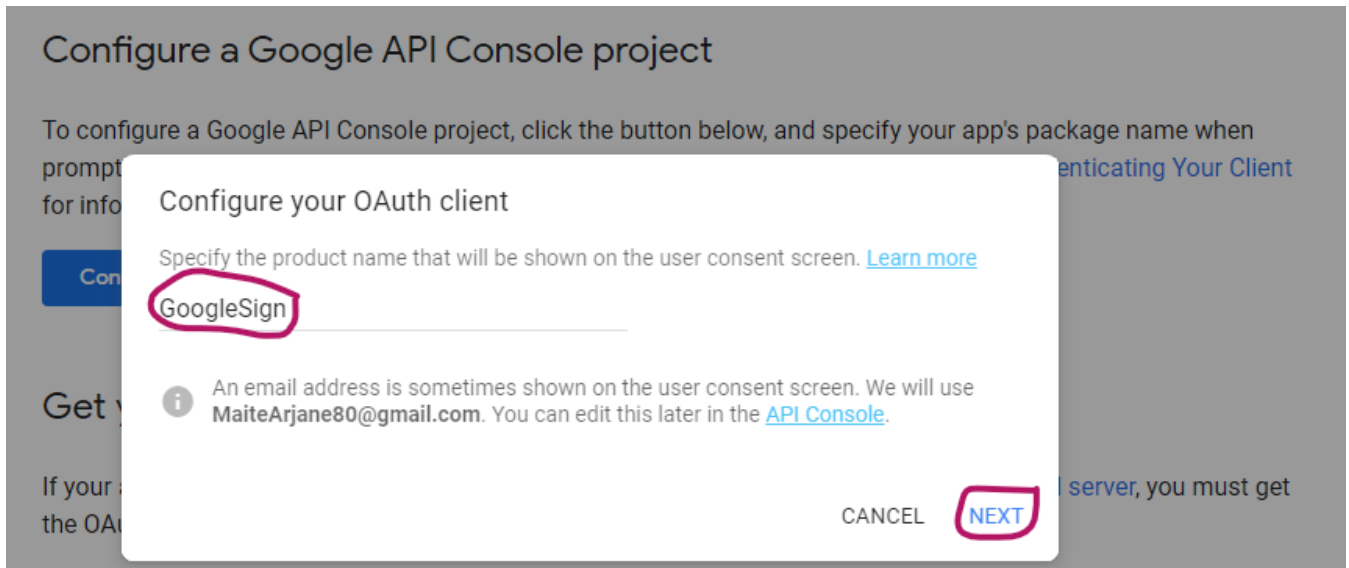
<https://developers.google.com/identity/sign-in/android/start-integrating>



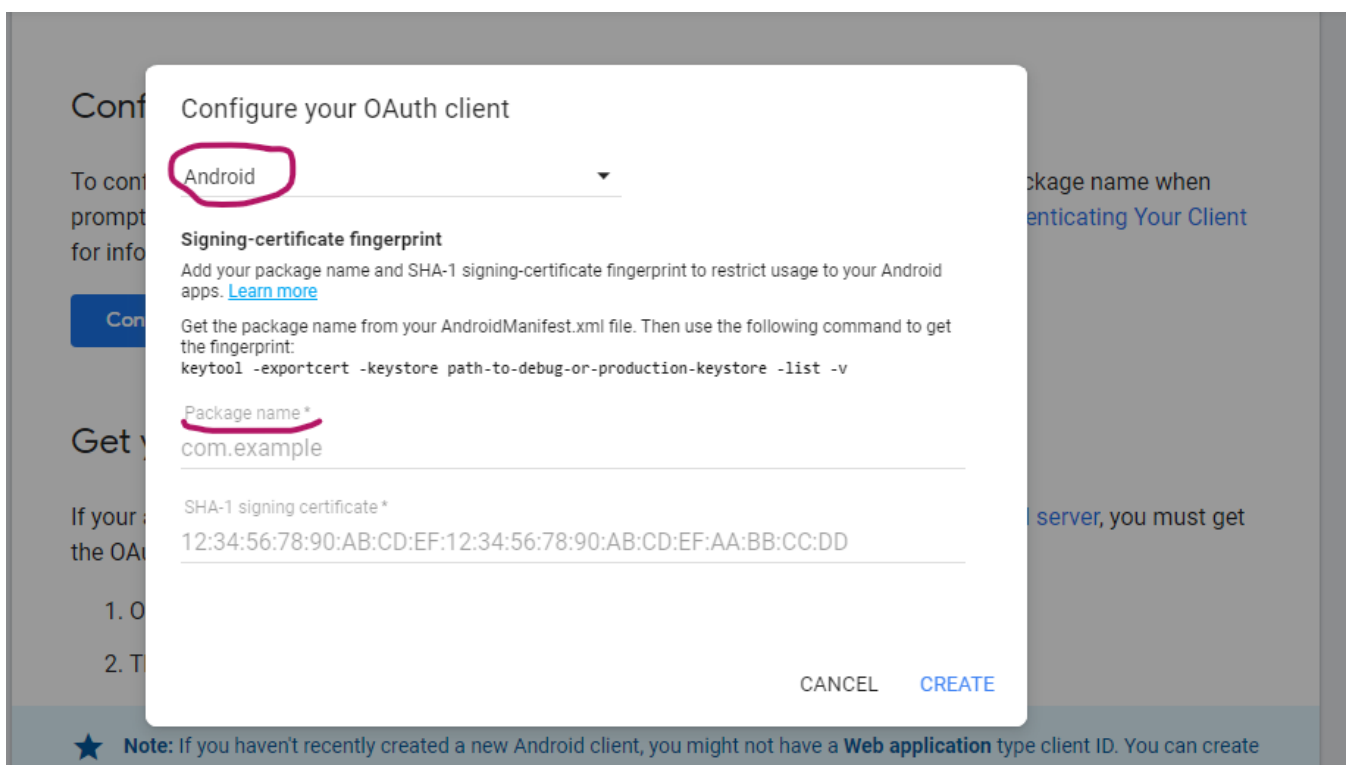
- On crée un nouveau projet.



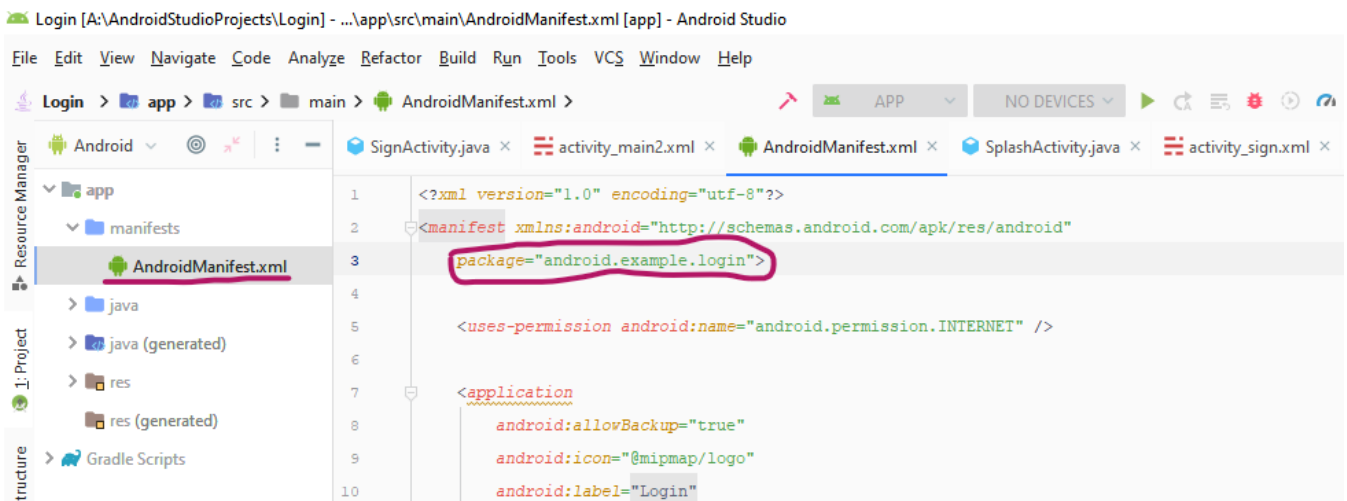
- Nous ajoutons un nom de produit



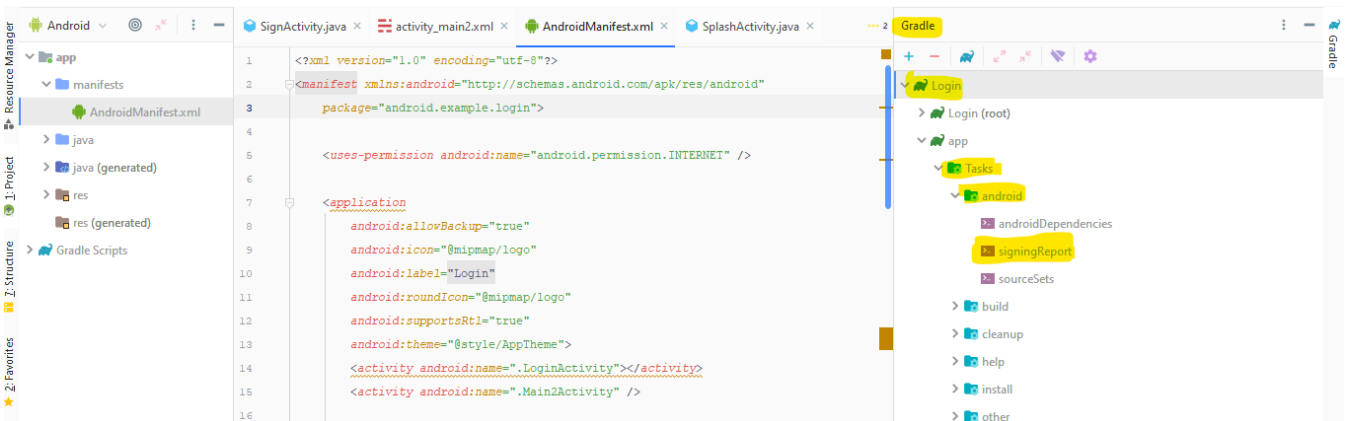
- Nous précisons le type d'application (Android) et le nom de package de l'application qui est mentionné dans le fichier AndroidManifest.xml



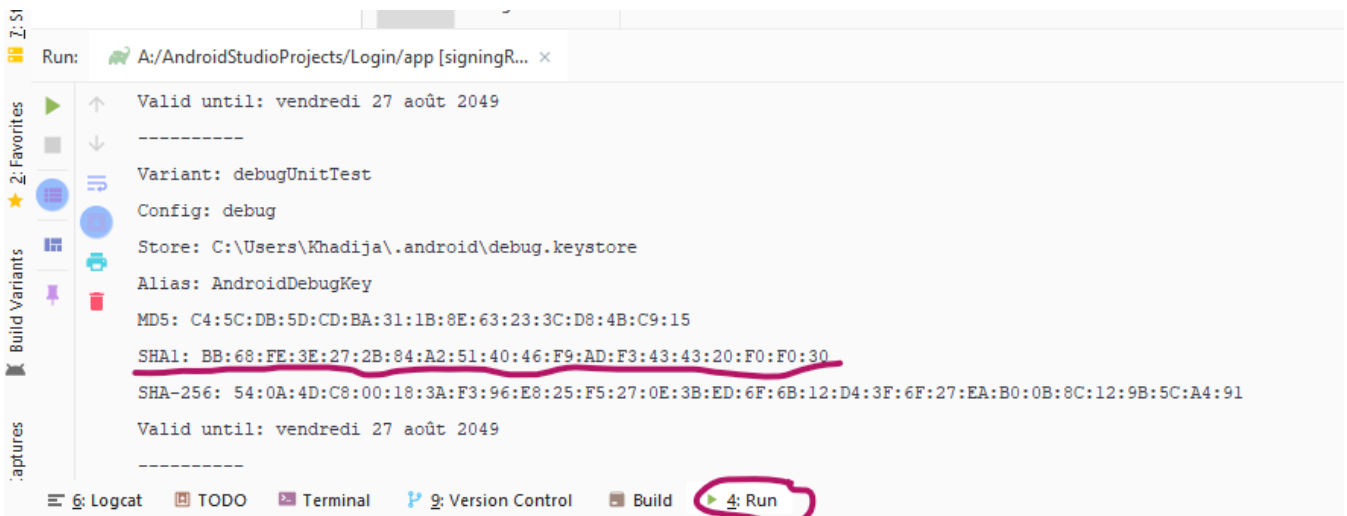
- **Le fichier AndroidManifest.xml**



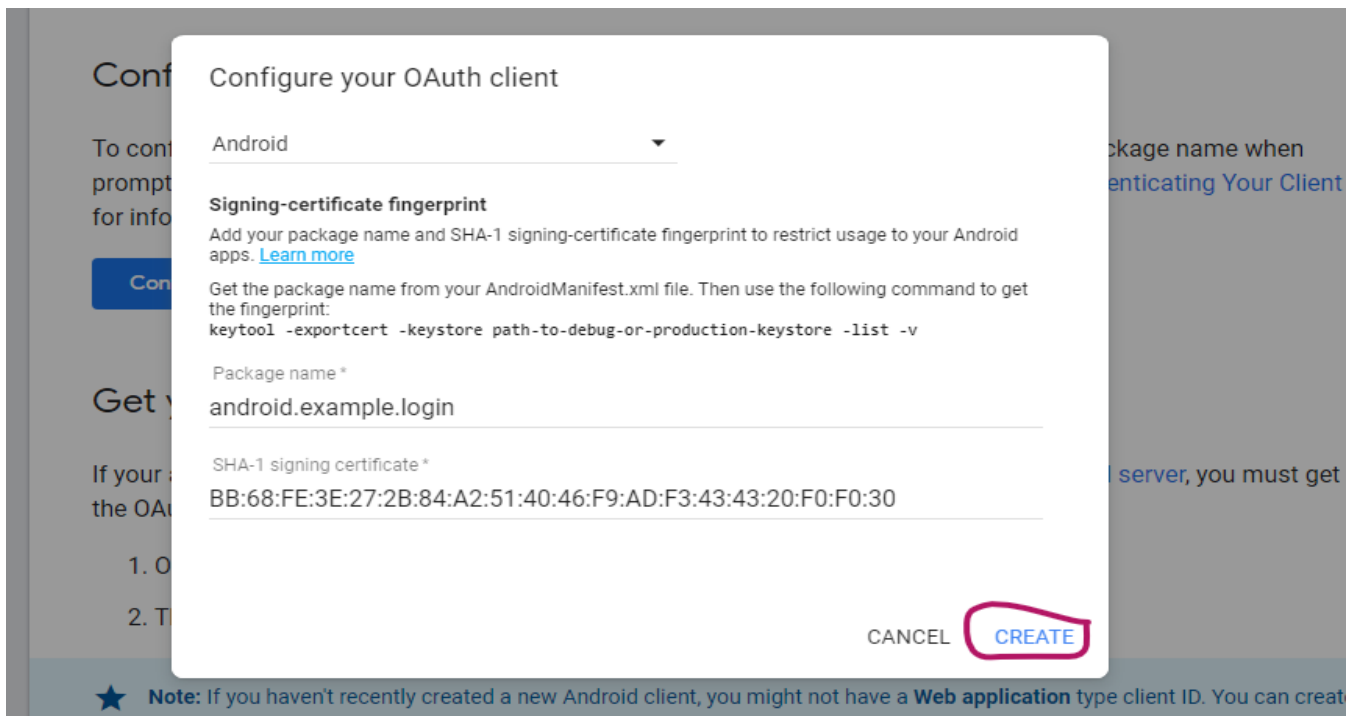
- Pour récupérer le Certificat de signature SHA-1
- On ouvre la fenêtre à droite de gradle et on trouve la structure suivante :



- On double clique sur la commande signingreport, et on récupère la clé SHA1 dans la console.



- On l'ajoute dans la configuration est c'est juste.



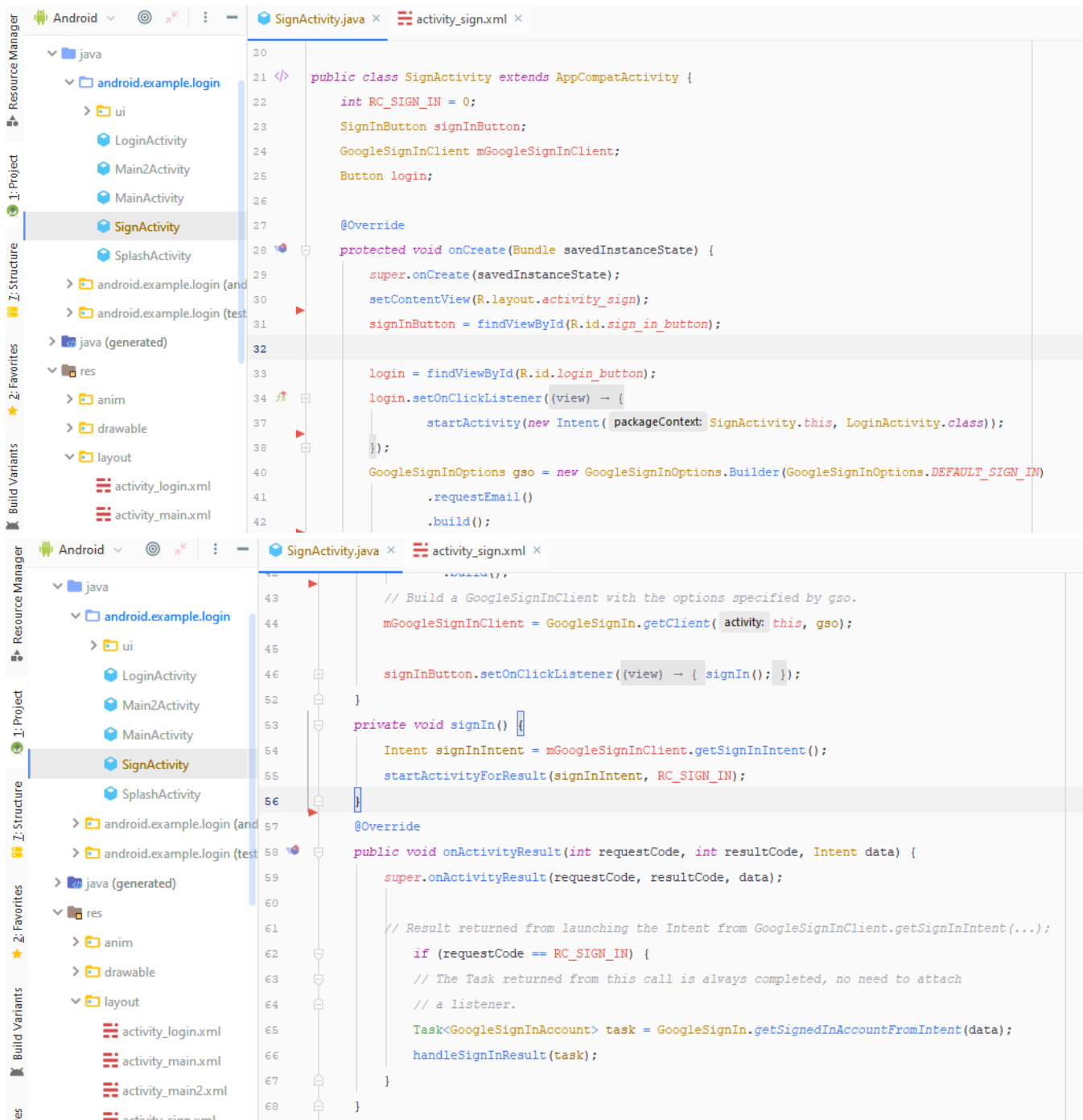
2.2. Intégration d'authentification via Google dans Notre Application Android

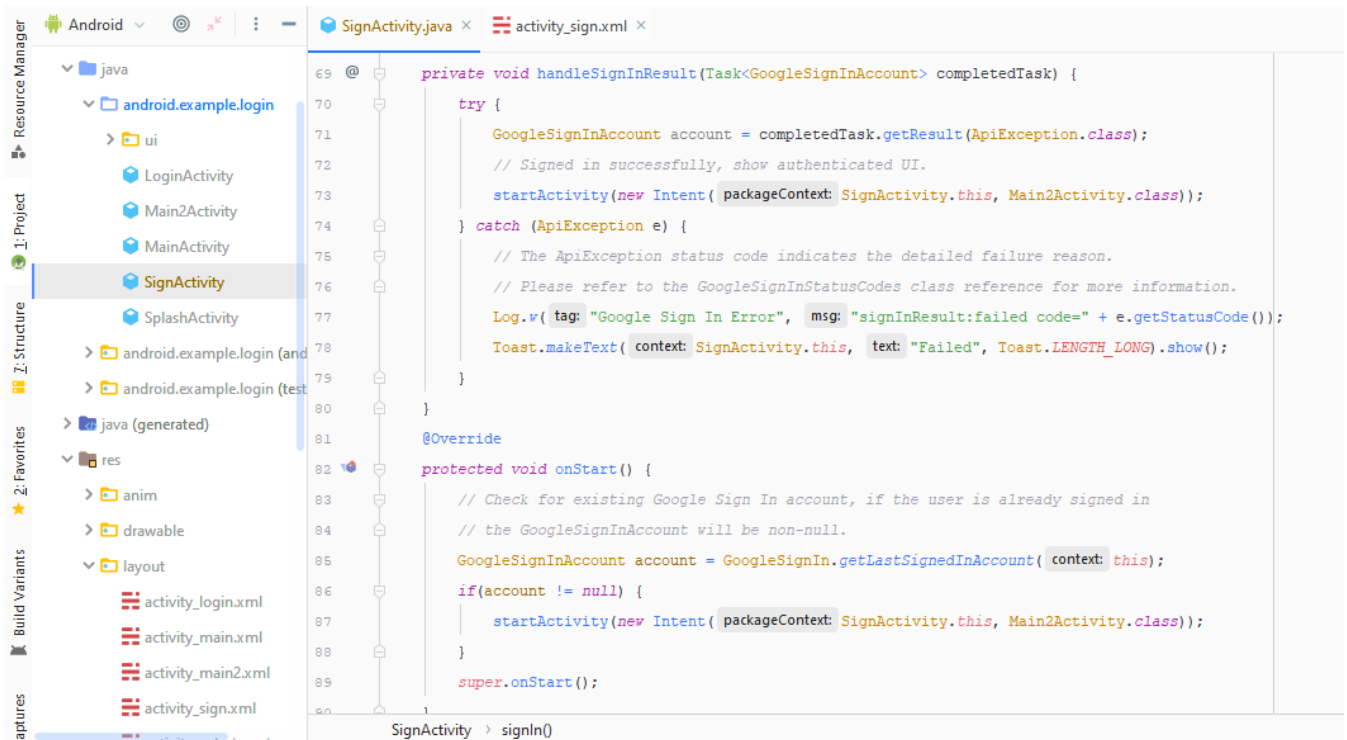
On ajoute une nouvelle activité (SignInActivity) avec un fichier xml (activity_Sign.xml).

- On ajoute le bouton de s'authentifier de Google.

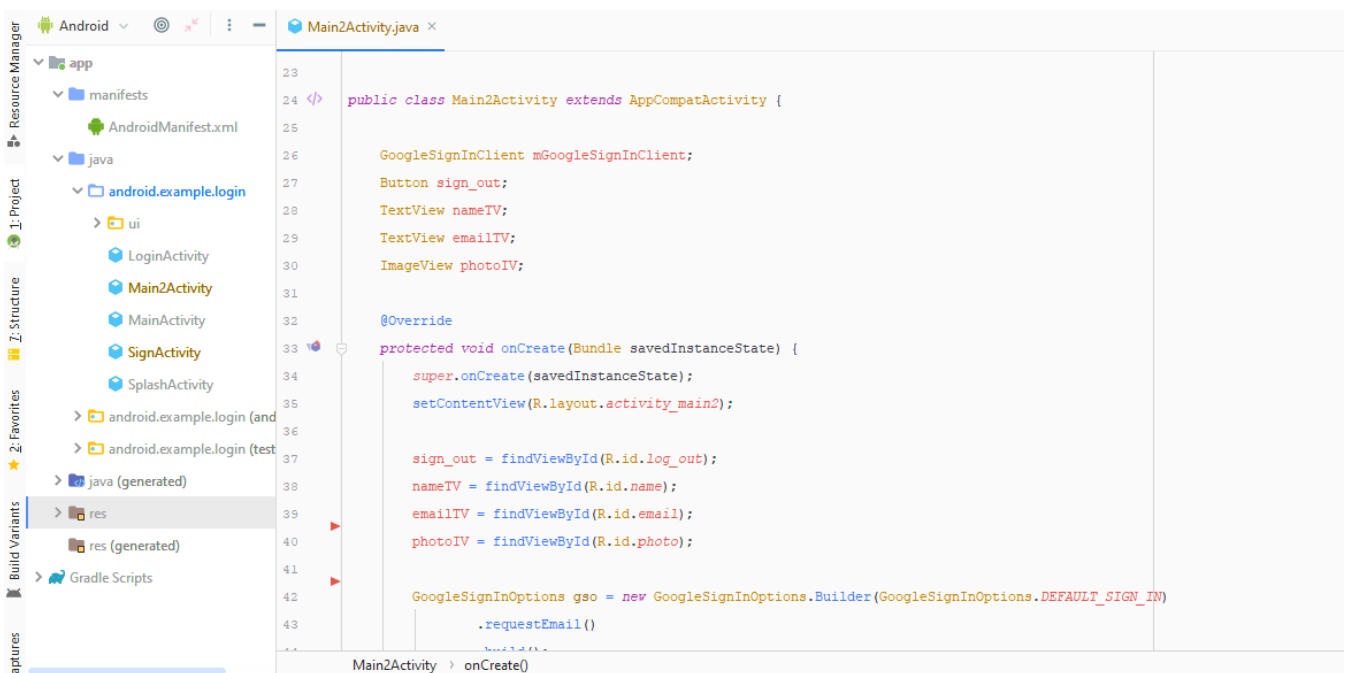


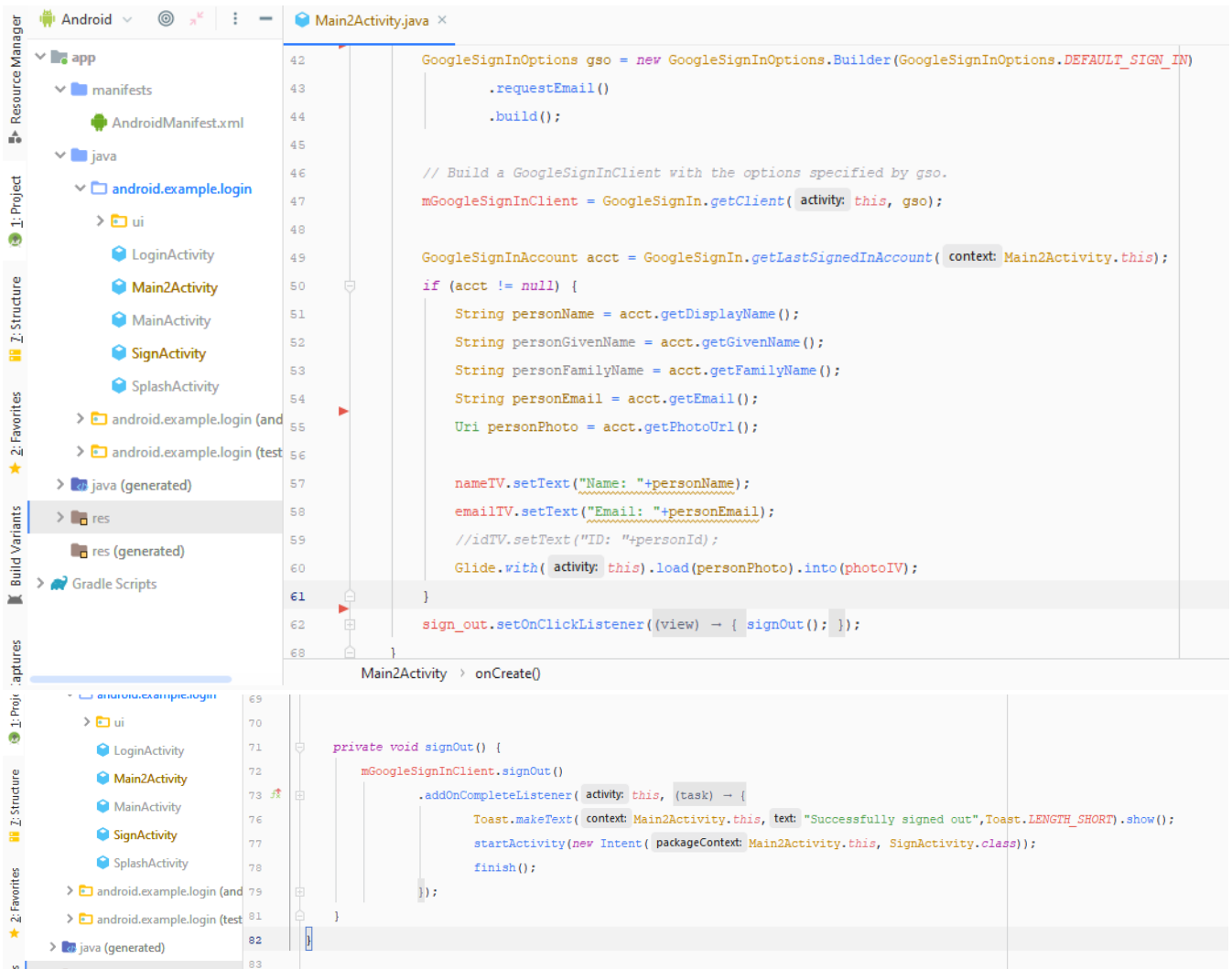
- Dans l'activité on ajoute tous les services qui permet de récupérer le compte gmail de client.



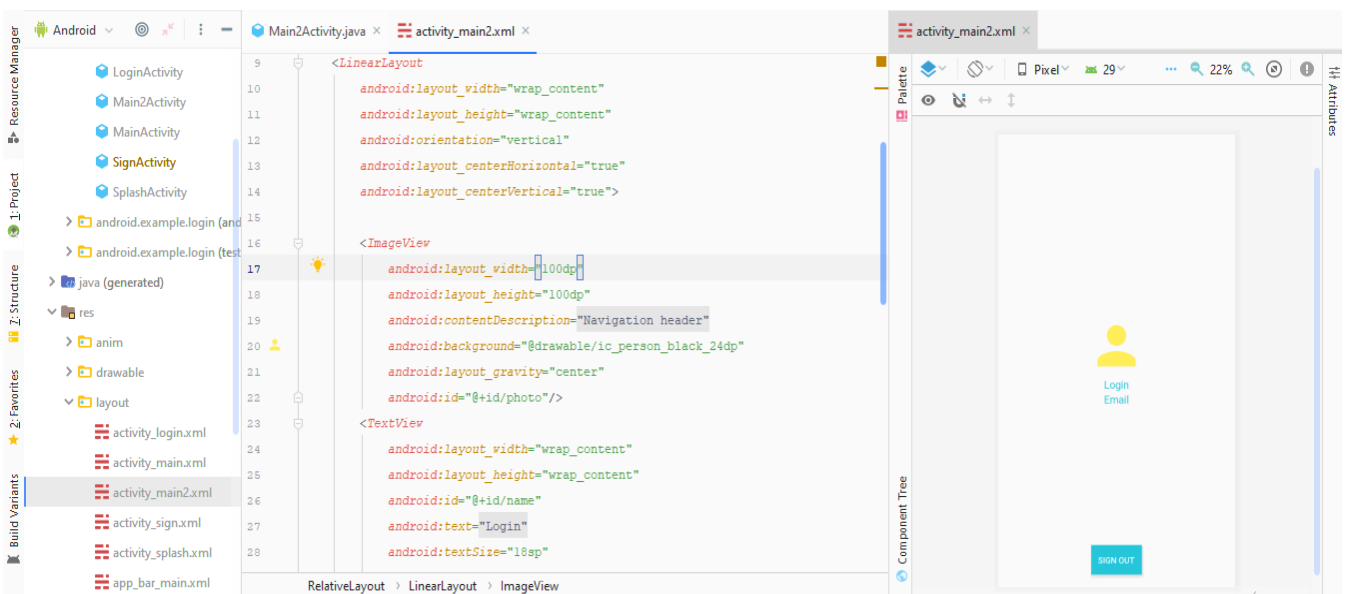


- Après on ajoute une nouvelle activity (Main2Activity) pour afficher les coordonnées de client.





- Dans le fichier activity_main2.xml on ajoute les textview pour afficher les données récupérées.



II. Ajouter Facebook Login à l'application Android

Les méthodes de connexion les plus couramment utilisées sont en utilisant la connexion Google ou en utilisant la connexion Facebook.

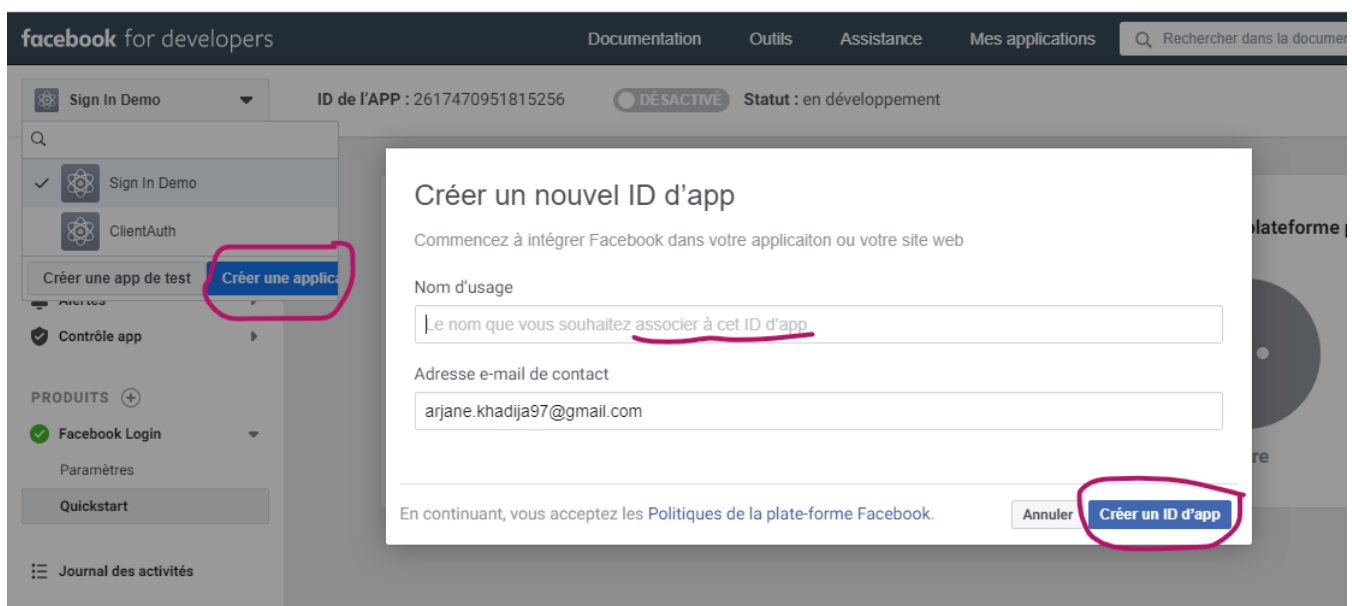
La façon la plus simple de configurer Facebook Login pour Android est via la documentation d'assistance fournie par Facebook. Allez dans Docs-> Facebook Login-> Android. Sélectionnez l'application nouvellement créée dans la section Sélectionner une application du document.

1. Créer une nouvelle application dans la console développeur de Facebook

Accédez à la console développeur de Facebook et créez une nouvelle application.
<https://developers.facebook.com/>

1.1. Générer votre ID d'application

- Indiquez le nom d'affichage de l'application et l'adresse e-mail de contact pour générer votre ID d'application.
- Spécifiez le nom d'affichage de votre application et l'adresse e-mail de contact

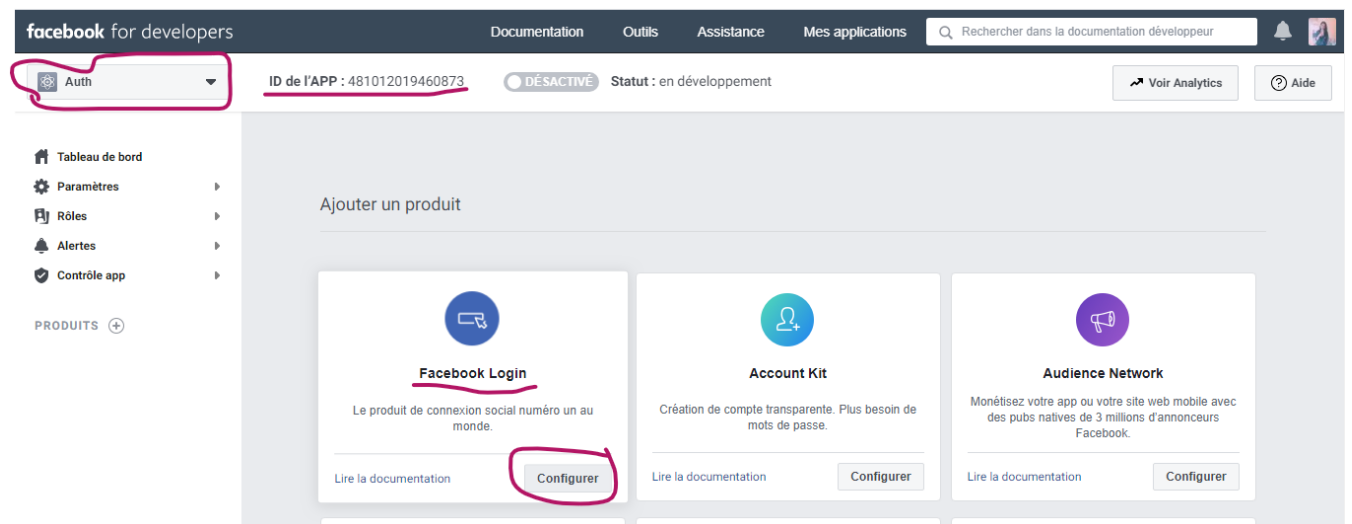


The screenshot shows the Facebook Developers console interface. On the left, a sidebar contains navigation links: 'Sign In Demo', 'ClientAuth', 'Créer une app de test', 'Contrôle app', 'PRODUITS', 'Facebook Login', 'Paramètres', 'Quickstart', and 'Journal des activités'. The main area displays the 'Créer un nouvel ID d'app' dialog. The dialog title is 'Créer un nouvel ID d'app'. Below the title, it says 'Commencez à intégrer Facebook dans votre application ou votre site web'. There are two input fields: 'Nom d'usage' with the placeholder text 'Le nom que vous souhaitez associer à cet ID d'app', and 'Adresse e-mail de contact' with the value 'arjane.khadija97@gmail.com'. At the bottom of the dialog, there is a checkbox for 'En continuant, vous acceptez les Politiques de la plate-forme Facebook.' and two buttons: 'Annuler' and 'Créer un ID d'app'. The 'Créer un ID d'app' button is highlighted with a red circle. In the background, the console shows the 'ID de l'APP : 2617470951815256' and a status of 'en développement'.

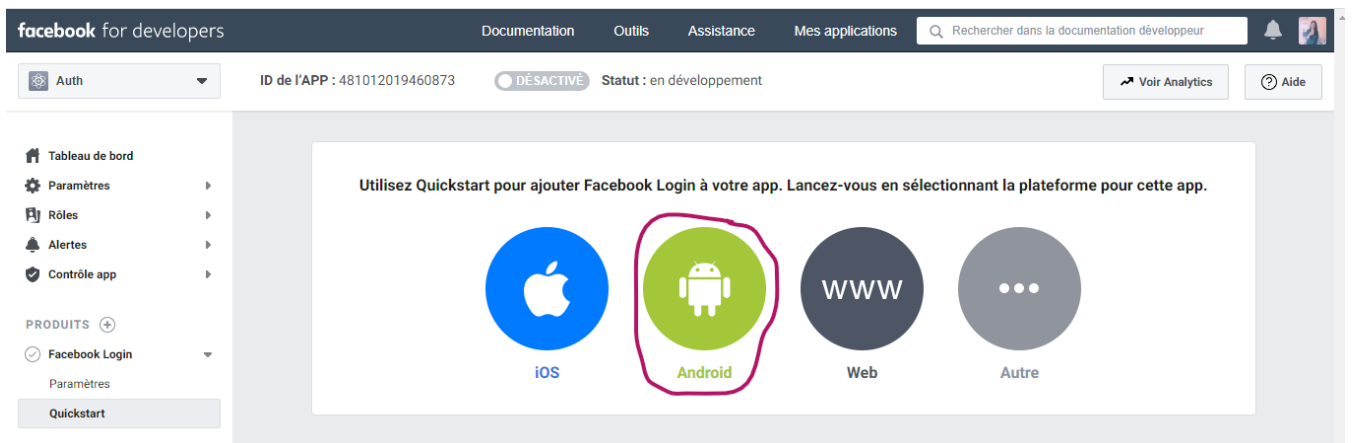
- Contrôle de sécurité



1.2. Configurer Facebook login



1.3. Spécifier le type de votre application



1.4. Ajouter les dépendances

L'étape suivante consiste à ajouter le SDK Facebook au projet Android Studio. Ouvrez **your_app > Gradle Scripts > build.gradle (Project)** et assurez-vous que les référentiels **jcenter** et **mavenCentral** sont répertoriés dans la section **buildscript > repositories**.

ID de l'APP : 481012019460873

☐ DÉSACTIVÉ Statut : en développement

Voir Analytics

3. Dans votre projet, ouvrez **your_app | Gradle Scripts | build.gradle (Project)** et ajoutez le référentiel suivant à la section **buildscript { repositories {} }** pour télécharger le SDK à partir du référentiel Maven Central :

```
mavenCentral()
```

4. Dans votre projet, ouvrez **your_app | Gradle Scripts | build.gradle (Module: app)** et ajoutez le code de compilation suivant à la section **dependencies{}** pour compiler la version la plus récente du SDK :

```
implementation 'com.facebook.android:facebook-android-sdk:[4,5)'
```

✓

Lorsque vous utilisez le SDK Facebook, certains événements dans votre app sont automatiquement consignés et collectés pour Facebook Analytics, à moins de désactiver la consignation automatique des événements. Pour en savoir plus sur les informations collectées et sur la façon de désactiver la consignation automatique des événements, consultez la section [Consignation automatique des app events](#).

5. Créez votre projet.

6. Ajoutez les chaînes suivantes pour importer les packages SDK Facebook :

```
import com.facebook.FacebookSdk;
import com.facebook.appevents.AppEventsLogger;
```

Copier le code

1.5. Ajouter Le package de votre application

ID de l'APP : 481012019460873

☐ DÉSACTIVÉ Statut : en développement

Voir Analytics

3. Expliquer votre projet Android

Nom du lot

Le nom de votre lot permet d'identifier de façon unique votre app Android. Nous l'utilisons pour permettre aux gens de télécharger votre app dans Google Play s'ils ne l'ont pas installée. Vous le trouverez dans votre manifeste Android ou dans le fichier build.gradle de votre app.

com.example.myapp

Nom de la classe d'activité par défaut

Il s'agit du nom de la classe entièrement qualifiée de l'activité qui gère les liens profonds, comme `com.example.app.DeepLinkingActivity`. Nous l'utilisons lorsque nous créons un lien profond vers votre app à partir de l'app Facebook. Vous le trouverez également dans votre manifeste Android.

com.example.myapp.MainActivity

Save

1.6. Ajouter vos clés de hachage de développement et de publication

Pour vérifier l'authenticité des interactions entre votre app et Facebook, vous devez nous fournir la clé de hachage Android correspondant à votre environnement de développement. Si votre app a déjà été publiée, ajoutez également votre clé de hachage de publication.

Vous devez disposer d'une clé de hachage de développement unique pour chaque environnement de développement Android.

ID de l'APP : 481012019460873 DÉSACTIVÉ Statut : en développement

Windows

Vous aurez besoin des éléments suivants :

- Outil de gestion de clé et de certificat (**keytool**) du kit de développement Java
- Bibliothèque OpenSSL **openssl-for-windows** pour Windows de Google Code Archive

Pour générer une clé de hachage de développement, exécutez la commande suivante dans une invite de commande dans le dossier SDK Java :

```
keytool -exportcert -alias androiddebugkey -keystore "C:\Users\USERNAME\.android\debug.keystore" | "F"
```

[Copier le code](#)

Cette commande génère une clé de hachage unique composée de 28 caractères pour votre environnement de développement. Copiez et collez-la dans le champ ci-dessous. Vous devez fournir une clé de hachage de développement pour l'environnement de développement de chaque personne qui travaille sur votre app.

- Télécharger Openssl sous Windows via ce lien

https://code.google.com/archive/p/openssl-for-windows/downloads?fbclid=IwAR0fgPsc7u0d71Xg11oxbwdkCIks_3B2TDS_bg-GFdgiTi7LGZWGyYZbWmlQ

Google Code Archive

Projects Search About


Project

Source

Issues

Wikis

Downloads

 **openssl-for-windows**

File	Summary + Labels	Uploaded	Size
openssl-0.9.8k_WIN32.zip	openssl-0.9.8k WIN32 Featured	Jul 23, 2009	1.25MB
openssl-0.9.8k_X64.zip	openssl-0.9.8k X64 Featured	Jul 23, 2009	1.42MB
openssl-0.9.8e_X64.zip	openssl-0.9.8e X64	Jul 23, 2009	1.25MB
openssl-0.9.8e_WIN32.zip	openssl-0.9.8e WIN32	Jul 23, 2009	1.08MB
openssl-0.9.8d_X64.rar	openssl-0.9.8d X64	Jul 23, 2009	1.32MB
openssl-0.9.8d_WIN32.rar	openssl-0.9.8d win32	Jul 23, 2009	1.16MB

- Après on tape la commande suivnt dans Cmd :

`keytool -exportcert -alias androiddebugkey -keystore "C:\Users\USERNAME\.android\debug.keystore" | "PATH_TO_OPENSSL_LIBRARY\bin\openssl" sha1 -binary | "PATH_TO_OPENSSL_LIBRARY\bin\openssl" base64`

```
C:\Program Files\Java\jdk1.8.0_221\bin>keytool -exportcert -alias androiddebugkey -keystore "C:\Users\Khadija\.android\debug.keystore" | "C:\openssl\bin\openssl" sha1 -binary | "C:\openssl\bin\openssl" base64
Entrez le mot de passe du fichier de clés : android

Warning:
Le fichier de clés JKS utilise un format propriétaire. Il est recommandé de migrer vers PKCS12, qui est un format standard de l'industrie en utilisant "keytool -importkeystore -srckeystore C:\Users\Khadija\.android\debug.keystore -destkeystore C:\Users\Khadija\.android\debug.keystore -deststoretype pkcs12".
u2j+PicrhKJRQEb5rfNDQyDw8DA=
C:\Program Files\Java\jdk1.8.0_221\bin>
```

- Par la suite on l'ajoute ici

Cette opération permet de générer une chaîne composée de 28 caractères que vous devez copier et coller dans le champ ci-dessous. Reportez-vous également à la [documentation Android](#) pour savoir comment signer vos apps.

Clés de hachage

Ex : nm0blrXpAM3cUshel

Save

1.7. Activer l'authentification unique pour votre app

5. Activer l'authentification unique pour votre app

Activer l'authentification unique

Si vous souhaitez que vos Notifications Android puissent lancer votre app, activez l'authentification unique.

☒ Oui

Authentification unique

Sera lancé à partir des applications Android

Save

Retour Suivant

2. Configurer notre application Android

2.1. Modifier vos ressources et votre manifeste

ID de l'APP : 481012019460873

☐ DÉSACTIVÉ Statut : en développement

6. Modifier vos ressources et votre manifeste

Créez des chaînes pour votre ID d'app Facebook et pour ceux qui sont nécessaires à l'activation des onglets personnalisés de Chrome. De plus, ajoutez `FacebookActivity` à votre manifeste Android.

1. Ouvrez votre fichier `/app/res/values/strings.xml`.
2. Ajoutez ce qui suit :

```
<string name="facebook_app_id">481012019460873</string> <string name="fb_login_protocol_scheme">fb</string>
```

3. Ouvrez le fichier `/app/manifest/AndroidManifest.xml`.
4. Ajoutez l'élément `uses-permission` suivant après l'élément `application` :

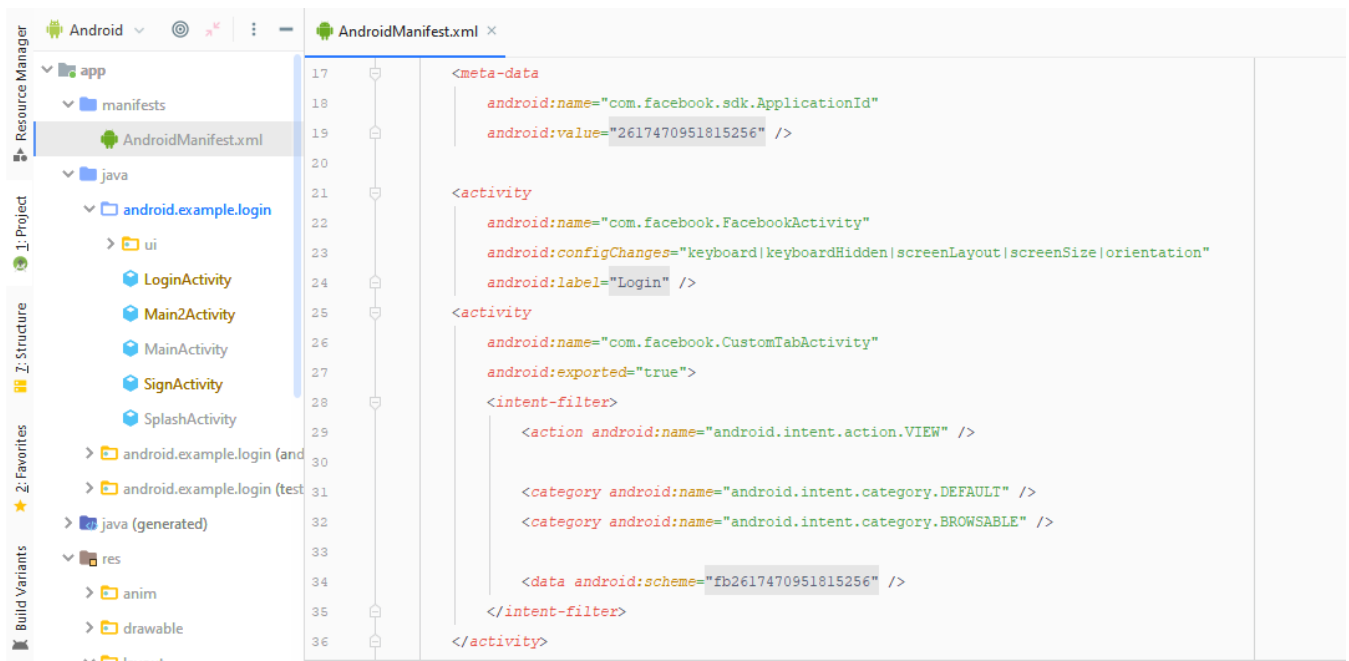
```
<uses-permission android:name="android.permission.INTERNET"/>
```

5. Ajoutez l'élément `meta-data` suivant, une activité pour Facebook, ainsi qu'une activité et filtre d'intentions pour les onglets personnalisés de Chrome à l'intérieur de votre élément `application` :

```
<meta-data android:name="com.facebook.sdk.ApplicationId" android:value="@string/facebook_app_id"/>  
<activity android:name="com.facebook.FacebookActivity" android:configChanges="keyboard|keyboardHidden|screenLayout|screenSize|orientation" android:label="@string/app_name" />
```

Retour Suivant

1. Internet Permission.
2. Meta-data element that mention your app Id.
3. Facebook Activity and intent filter for Chrome Custom Tabs.



2.2. Ajouter le bouton Facebook Login

Le moyen le plus simple d'ajouter Facebook Login à votre app est d'ajouter LoginButton à partir du SDK. LoginButton est un élément d'interface qui renferme la fonctionnalité disponible dans LoginManager. Lorsque quelqu'un clique sur ce bouton, le processus de connexion est lancé avec les autorisations définies dans LoginManager. Le bouton s'accorde avec l'état de connexion, et affiche le message correspondant à l'état d'authentification de la personne.

Pour ajouter le bouton Facebook Login, vous devez d'abord l'ajouter à votre fichier de disposition XML :

```
<com.facebook.login.widget.LoginButton
    android:id="@+id/login_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:layout_marginTop="30dp"
    android:layout_marginBottom="30dp" />
```

Copier le code

Retour Suivant

The screenshot displays the Android Studio interface. The top editor shows the 'activity_login.xml' file with the following XML code:

```
<Button
    android:id="@+id/login_button"
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:layout_marginTop="30dp"
    android:background="@color/colorPrimary"
    android:text="Facebook"/>
```

The bottom editor shows the 'activity_login.xml' file with the following XML code:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".LoginActivity"
    android:orientation="vertical"
    android:gravity="center">
    <de.hdodenhof.circleimageview.CircleImageView
        android:layout_width="175dp"
        android:layout_height="175dp"
        android:layout_gravity="center_horizontal"
        android:id="@+id/profile_pic"
        />
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
```

The right sidebar shows the 'Component Tree' with a 'Continue with Facebook' button visible.

2.3. Enregistrer un rappel

Ensuite, créez un callbackManager chargé de gérer les réponses aux demandes de connexion en appelant `CallbackManager.Factory.create`.

```
callbackManager = CallbackManager.Factory.create();
```

Copier le code

Si vous ajoutez le bouton à un Fragment, vous devez également mettre à jour votre activité pour utiliser votre fragment. Vous pouvez personnaliser les propriétés de `Login button` et enregistrer un rappel dans votre méthode `onCreate()` ou `onCreateView()`. Les propriétés que vous pouvez personnaliser comprennent `LoginBehavior`, `DefaultAudience`, `ToolTipPopup.Style` et les autorisations associées à `LoginButton`. Par exemple :

```
loginButton = (LoginButton) findViewById(R.id.login_button);
loginButton.setReadPermissions("email");
// If using in a fragment
loginButton.setFragment(this);

// Callback registration
```

Copier le code

Pour répondre au résultat d'une demande de connexion, vous devez enregistrer un rappel soit avec `LoginManager`, soit avec `LoginButton`. Si vous enregistrez le rappel avec `LoginButton`, vous n'avez pas besoin d'enregistrer le rappel dans le gestionnaire de connexion.

Dans le fichier `LoginActivity.java`, nous faisons la logique de code suivante :

- Demander des autorisations de messagerie et de profil public sur le bouton de connexion Facebook.
- Pour gérer la réponse de connexion, créez un objet `CallbackManager` à l'aide de `Factory.create`
- Pour répondre au résultat de la connexion, nous enregistrons un rappel avec `LoginButton`.
- Si la connexion réussit, le paramètre `LoginResult` de la méthode de rappel `onSuccess ()` a un nouveau `AccessToken` et les autorisations accordées et refusées les plus récentes.
- Dans cet exemple, pour obtenir la réponse de connexion, nous n'utiliserons pas les méthodes de rappel avec `LoginButton`, nous utilisons plutôt la classe `AccessTokenTracker`.
- Substituez la méthode `onActivityResult` pour transmettre le résultat de la connexion au `LoginManager` à l'aide de `callbackManager`.
- Pour vérifier l'état de la connexion lors du chargement de l'application, appelez la méthode `getCurrentAccessToken ()` à l'aide de la classe `AccessToken` et si la méthode renvoie null signifie que l'utilisateur s'est déconnecté.
- Pour lire les informations utilisateur d'`AccessToken`, nous utilisons l'API Graph. Facebook utilise une API spéciale appelée API Graph pour lire et écrire des informations sur la plateforme Facebook. L'API Graph apporte le résultat sous la forme d'un objet JSON ou d'un tableau JSON.

```

31 <?xml version="1.0" encoding="utf-8"?>
32 public class LoginActivity extends AppCompatActivity {
33     private LoginButton loginButton;
34     private CircleImageView circleImageView;
35     private TextView txtName,txtEmail;
36
37     private CallbackManager callbackManager;
38     @Override
39     protected void onCreate(Bundle savedInstanceState) {
40         super.onCreate(savedInstanceState);
41         setContentView(R.layout.activity_login);
42         loginButton = (LoginButton) findViewById(R.id.login_button);
43         txtName = findViewById(R.id.profile_name);
44         txtEmail = findViewById(R.id.profile_email);
45         circleImageView = findViewById(R.id.profile_pic);
46         callbackManager = CallbackManager.Factory.create();
47         loginButton.setReadPermissions(Arrays.asList("email","public_profile"));
48         checkLoginStatus();
49
50         loginButton.registerCallback(callbackManager, new FacebookCallback <LoginResult>() {
51             @Override
52             public void onSuccess(LoginResult loginResult) {}
53
54             @Override
55             public void onCancel() {}
56
57             @Override
58             public void onError(FacebookException error) {}
59         });
60     }

```

- Si la connexion aboutit, le paramètre LoginResult contient le nouveau AccessToken ainsi que les autorisations récemment accordées ou refusées.
- Vous n'avez pas besoin d'un registerCallback pour que la connexion aboutisse. Vous pouvez choisir de suivre les modifications du token d'accès actuel à l'aide de la classe AccessTokenTracker, tel qu'indiqué ci-dessous.
- Enfin, dans votre méthode onActivityResult, appelez callbackManager.onActivityResult pour transmettre les résultats de la demande de connexion à LoginManager via callbackManager.

```

61
62 @Override
63 protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
64     callbackManager.onActivityResult(requestCode,resultCode,data);
65     super.onActivityResult(requestCode, resultCode, data);
66 }

```

2.4. Vérifier l'état de la connexion

Votre app ne peut prendre en charge qu'une personne connectée à la fois.

- LoginManager se charge de paramétrer le AccessToken et le Profile actuels de cette personne.
- Le SDK Facebook enregistre ces données dans les préférences communes et se charge de les paramétrer au début de la session.
- Vous pouvez déterminer si quelqu'un est actuellement connecté en vérifiant AccessToken.getCurrentAccessToken() et Profile.getCurrentProfile().

Vous pouvez charger `AccessToken.getCurrentAccessToken()` avec le SDK à partir du cache ou d'un signet d'app au lancement de votre app à partir d'un démarrage à froid. Vous devez vérifier sa validité dans la méthode `onCreate` de votre `Activity` :

```
AccessToken accessToken = AccessToken.getCurrentAccessToken();
boolean isLoggedIn = accessToken != null && !accessToken.isExpired();
```

Copier le code

Vous pouvez alors procéder à la connexion ultérieurement, par exemple, avec `OnClickListener` et un bouton personnalisé :

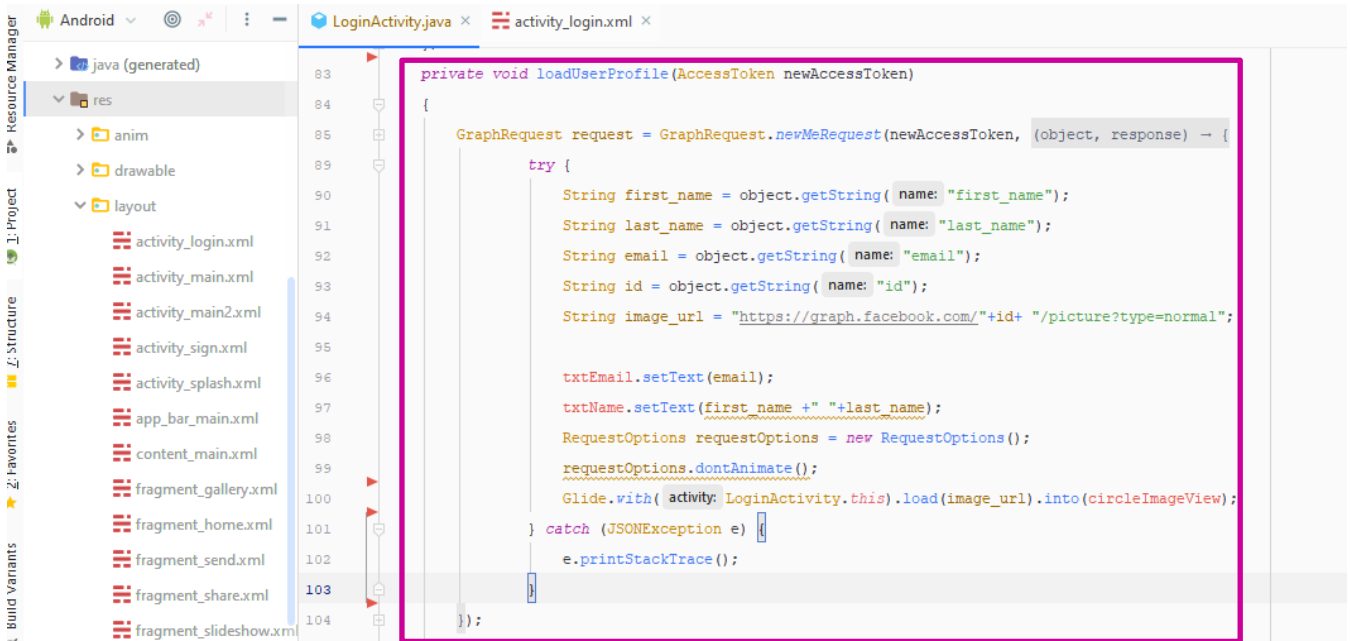
```
LoginManager.getInstance().loginWithReadPermissions(this, Arrays.asList("public_profile"));
```

Copier le code

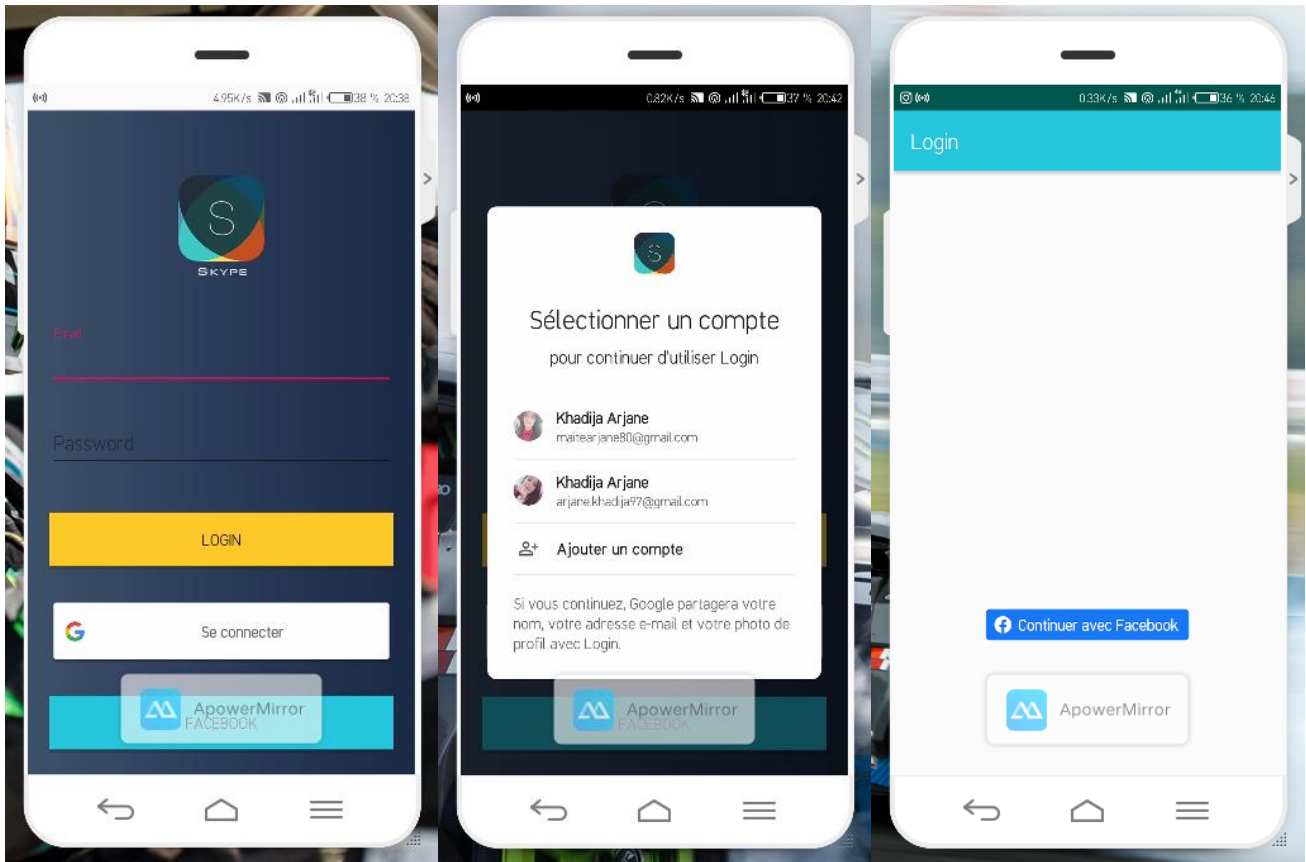
Retour

Suivant





III. Démonstration



IV. Conclusion

Nous avons réussi d'intégrer le bouton de connexion Facebook à l'application Android. J'espère que vous comprenez le concept.

Vous trouvez le projet sous Github via ce lien :

https://github.com/maiteAr/Login_Facebook_Google_Android