

Android Studio : Les bases !



Powered by the IntelliJ Platform

Android Studio : Les bases !

- ▶ 1) Installer Android Studio (Emulateur + Device manager)
- ▶ 2) Démonstration de l'authentification + Envoie du token + réception de notification push !
- ▶ 3) Ajout du sdk facebook + bouton de partage !
- ▶ 4) Déployer son application sur le store !



Installation Android Studio (Device et Emulateur)

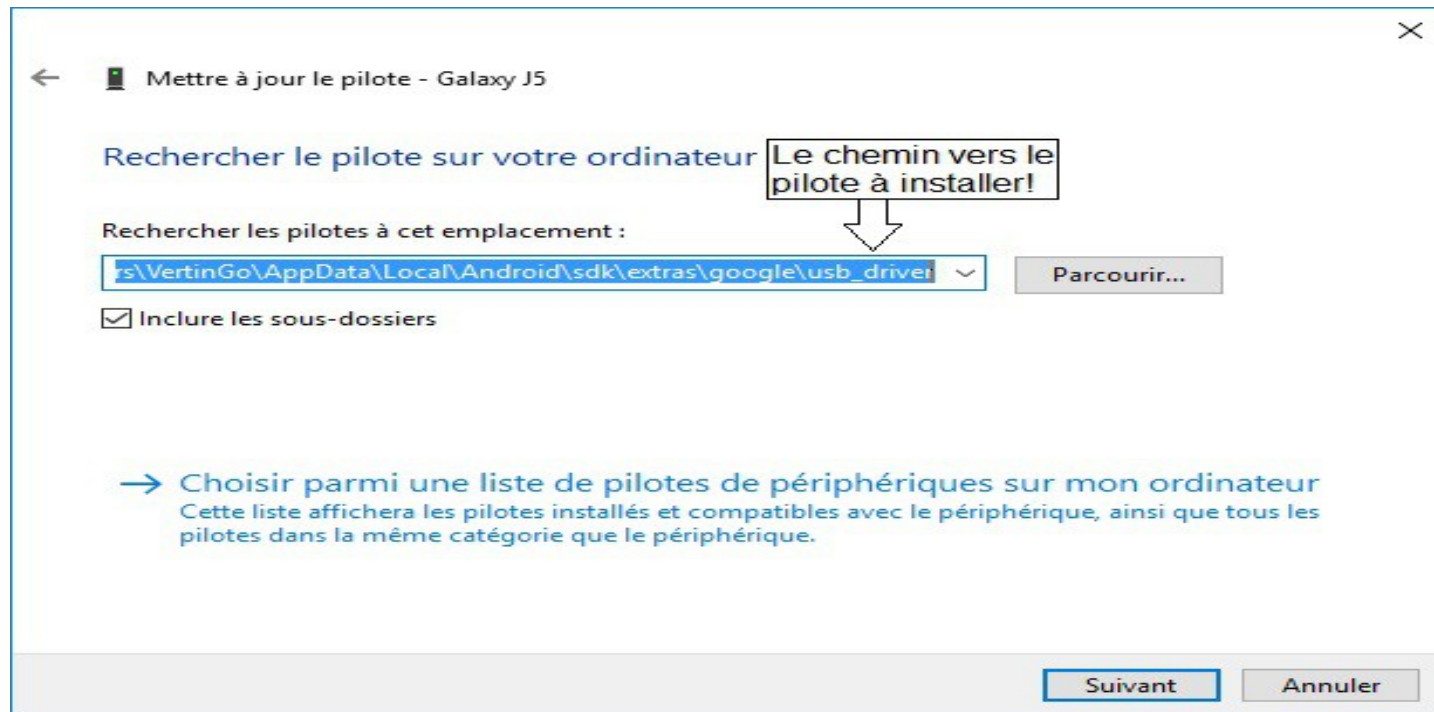
- ▶ Partie installation par le gestionnaire de périphériques pour envoyer votre programme sur le mobile !
- ▶ 1) Assurer vous d'avoir votre mobile relié à votre ordinateur par câble USB
- ▶ 2) Aller dans votre gestionnaire de périphériques (Device manager)
- ▶ 3) Sélectionner votre mobile et clic droit mettre à jour le pilote puis indiquer le chemin vers le driver usb_driver situé ci-dessous :

C:\Users\VertinGo\AppData\Local\Android\sdk\extras\google\usb_driver

- ▶ Ensuite choisir parmi une liste de pilotes sur mon ordinateur

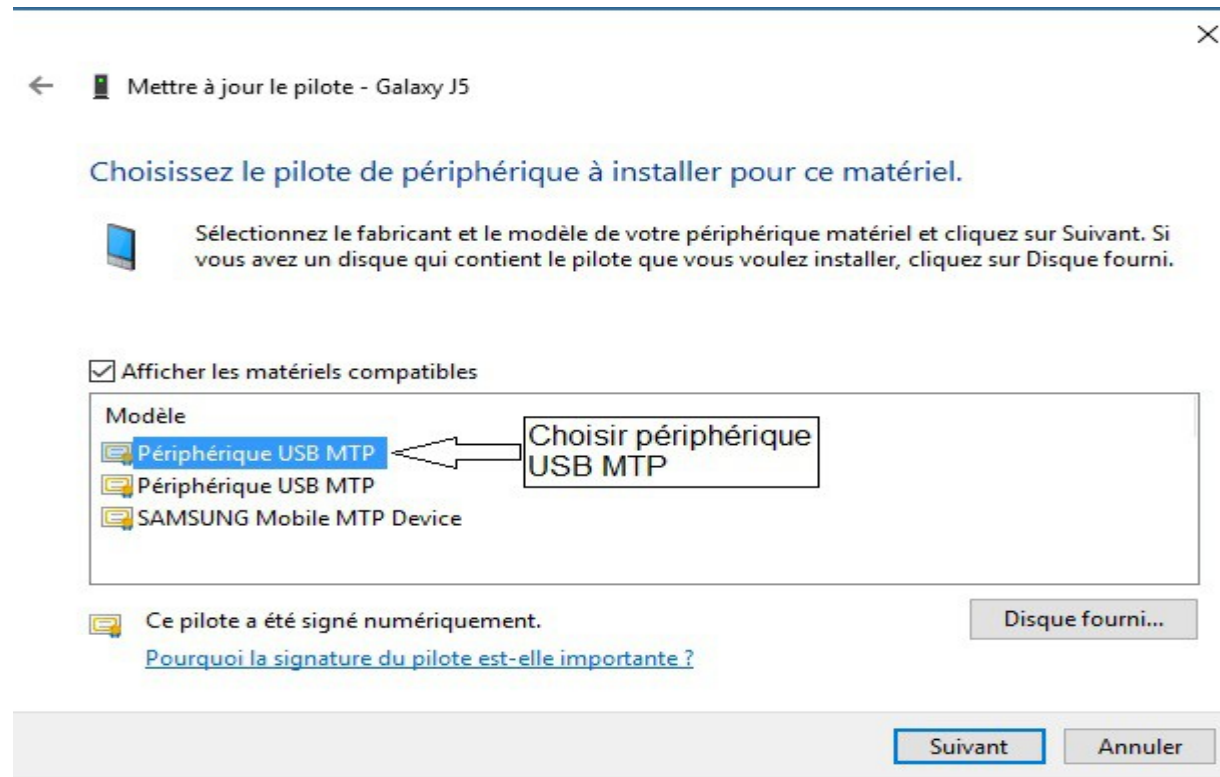
Installation Android Studio (Device et Emulateur)

- ▶ Une fois le lien vers votre pilote usb_driver renseigné cliquer sur « Choisir parmi une liste de pilotes de périphériques sur mon ordinateur »



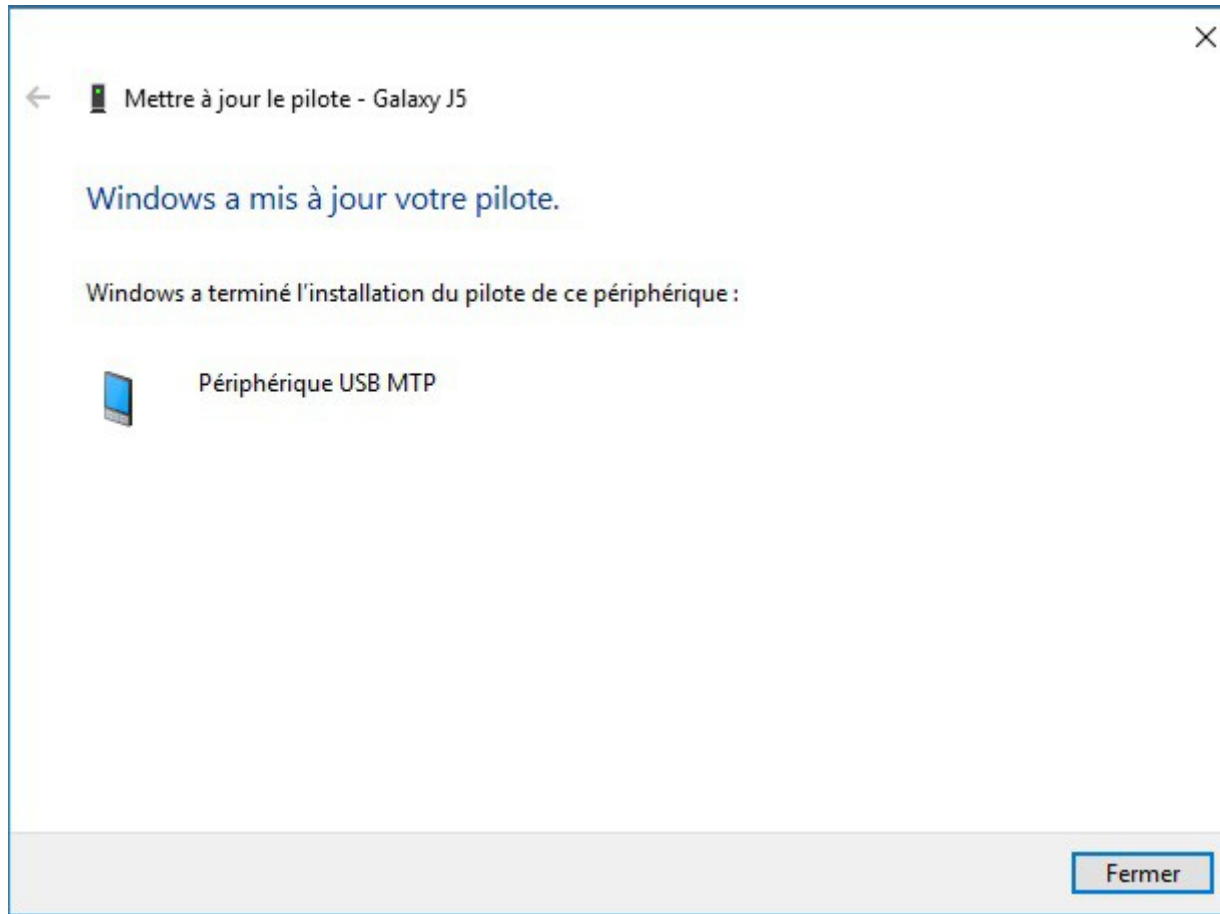
Installation Android Studio (Device et Emulateur)

- Choisir parmi les matériels compatibles le périphérique USB MTP



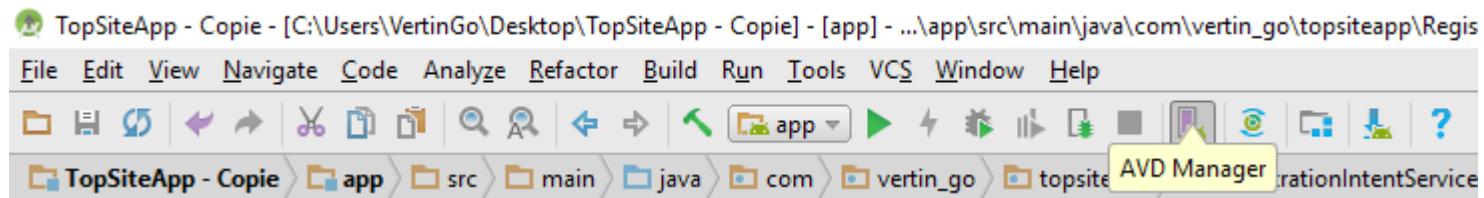
Installation Android Studio (Device et Emulateur)

► Fin de l'installation!



Installation Android Studio (Device et Emulateur)

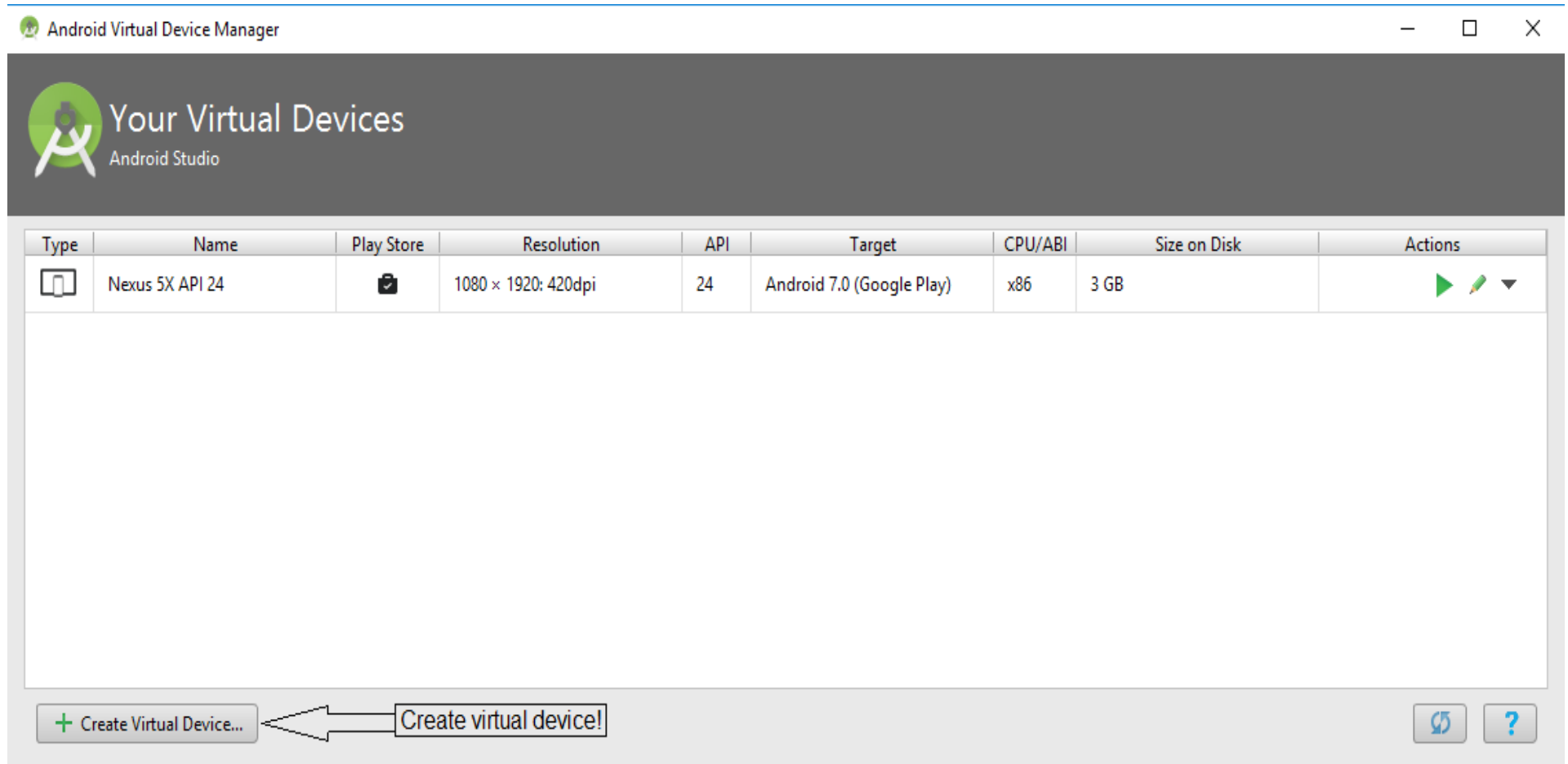
- ▶ Partie installation d'un périphérique virtuel avec Android Virtual Device Manager!
- ▶ Aller dans AVD Manager(Android Device Manager)



- ▶ Create Virtual Device !

Installation Android Studio (Device et Emulateur)

► Create Virtual Device!



Installation Android Studio (Device et Emulateur)

Virtual Device Configuration

Select Hardware
Android Studio

Choose a device definition

Choisir parmi les categories disponibles

Category	Name	Play Store	Size	Resolution	Density
TV	Pixel XL		5,5"	1440x2560	560dpi
Wear	Pixel		5,0"	1080x1920	xxhdpi
Phone	Nexus 5		4,0"	480x800	hdpi
Phone	Nexus One		3,7"	480x800	hdpi
Phone	Nexus 6P		5,7"	1440x2560	560dpi
Phone	Nexus 6		5,96"	1440x2560	560dpi
Phone	Nexus 5X		5,2"	1080x1920	420dpi
Phone	Nexus 5		4,95"	1080x1920	xxhdpi
Phone	Nexus 4		4,7"	768x1280	xhdpi
Tablet	Galaxy Nexus		4,65"	720x1280	xhdpi

New Hardware Profile Import Hardware Profiles

Nexus 5X


Size: large
Ratio: long
Density: 420dpi

Clone Device...

Cliquer sur next! Previous Next Cancel Finish Help

Installation Android Studio (Device et Emulateur)

Virtual Device Configuration

 **System Image**
Android Studio


Select a system image

Recommended x86 Images Other Images

Release Name	API Level	ABI	Target
Download	26	x86	Android 8.0 (Google Play)
Nougat	24	x86	Android 7.0 (Google Play)

Choisir parmi ceux recommandé!

Nougat



API Level
24

Android
7.0

Google Inc.

System Image
x86


We recommend these Google Play images because this device is compatible with Google Play.

Questions on API level?
See the [API level distribution chart](#)

Cliquer sur next! Previous Next Cancel Finish Help



Installation Android Studio (Device et Emulateur)

Virtual Device Configuration


 **Android Virtual Device (AVD)**
Android Studio


Verify Configuration

AVD Name

 Nexus 5X	5.2 1080x1920 xxhdpi	Change...
 Nougat	Android 7.0 x86	Change...

Startup orientation

 Portrait

 Landscape

Emulated Performance Graphics:

Device Frame ☒ Enable Device Frame

[Show Advanced Settings](#)

AVD Name

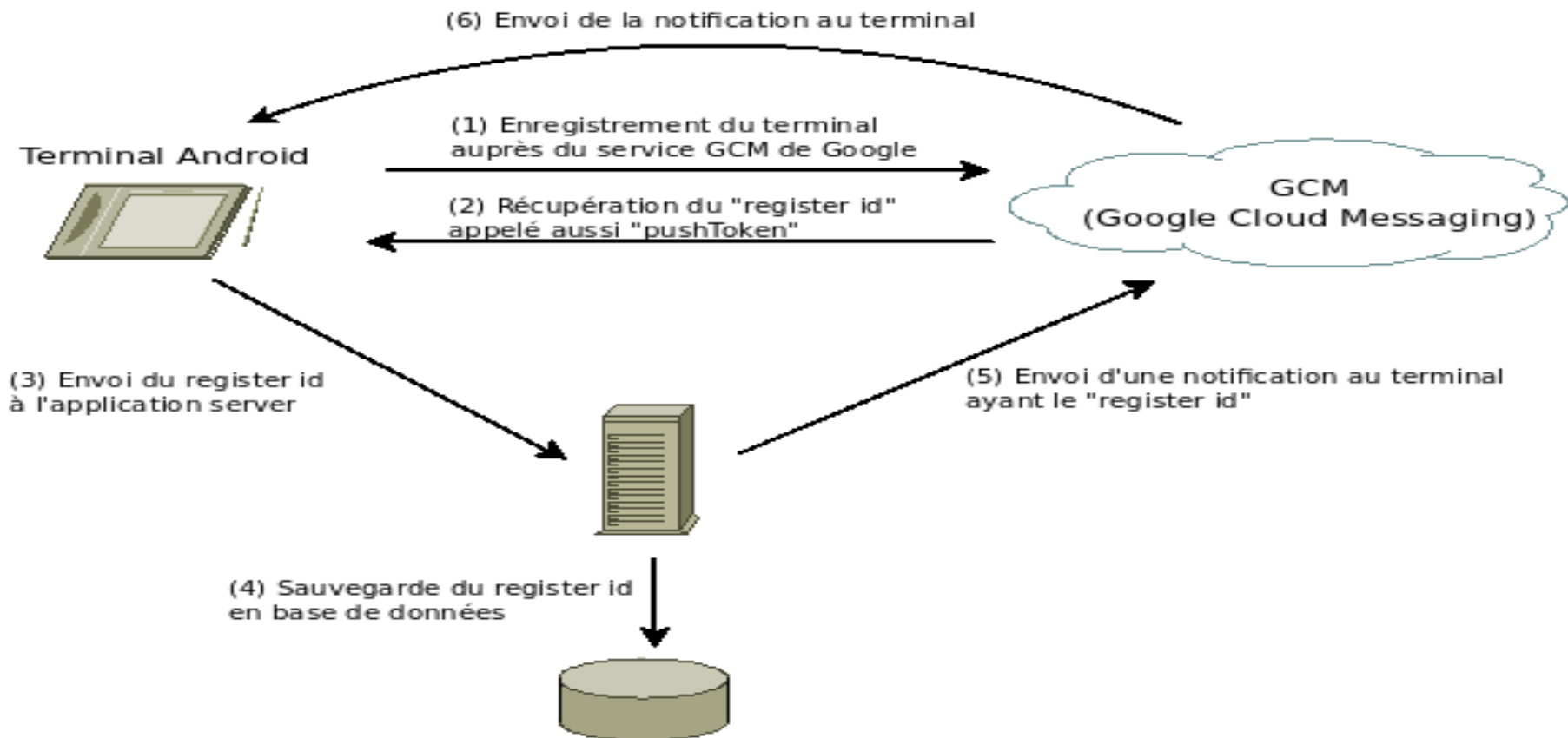
The name of this AVD.

Cliquer sur finish pour finaliser la configuration de votre device manager!

[Previous](#) [Next](#) [Cancel](#) [Finish](#) [Help](#)

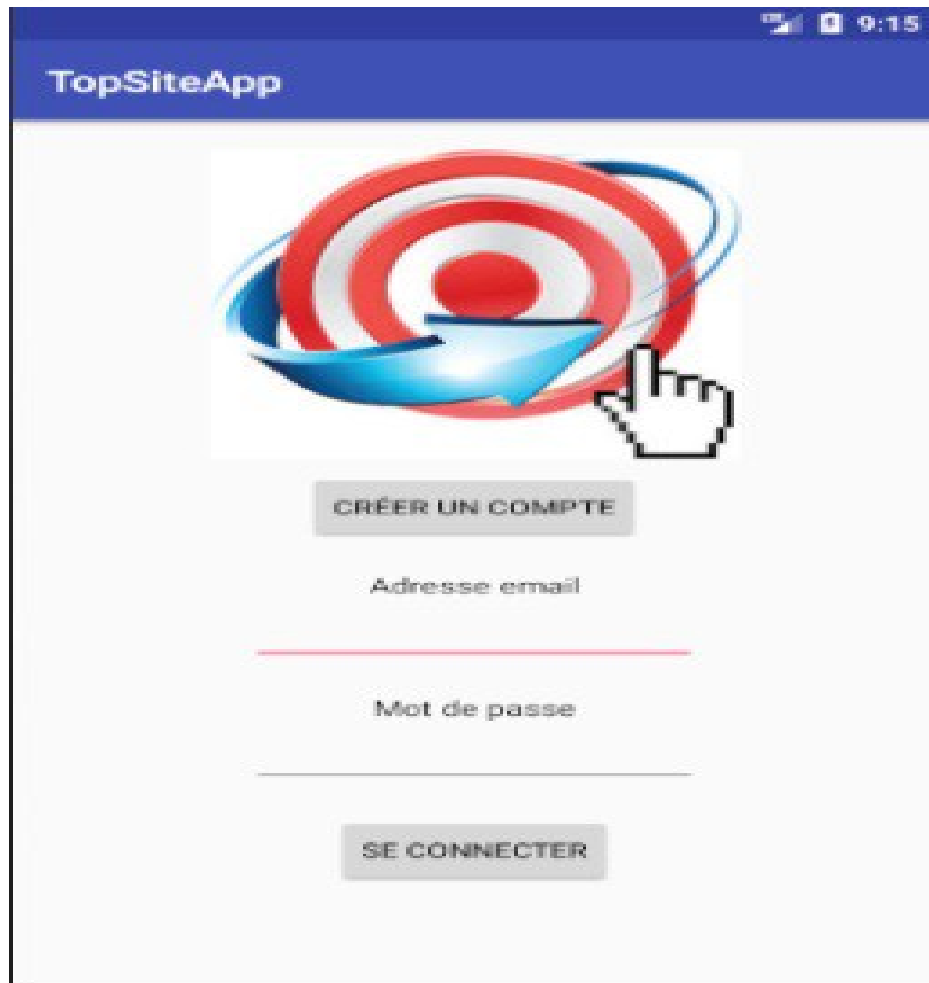
Système de notification push!

- Comment mettre en place un système de notification push côté serveur et client !

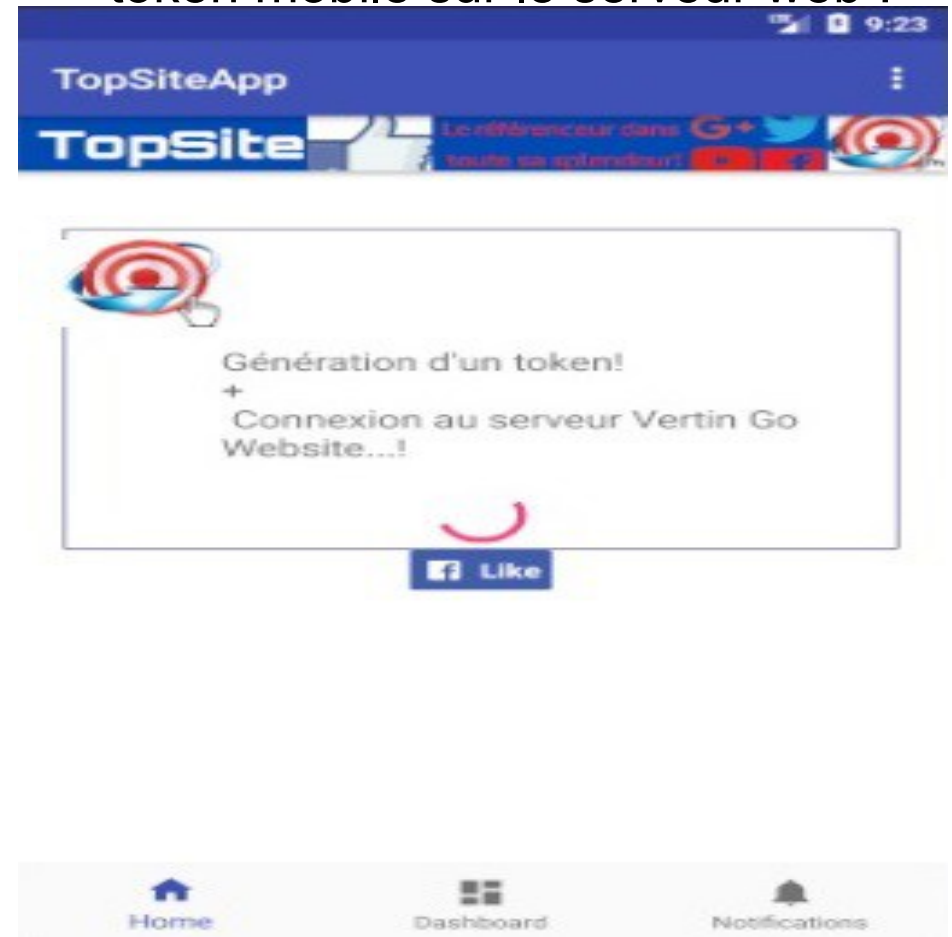


Présentation de l'application TopSite!

► Écran de connexion !



► Connexion en cours + envoi d'un token mobile sur le serveur web !



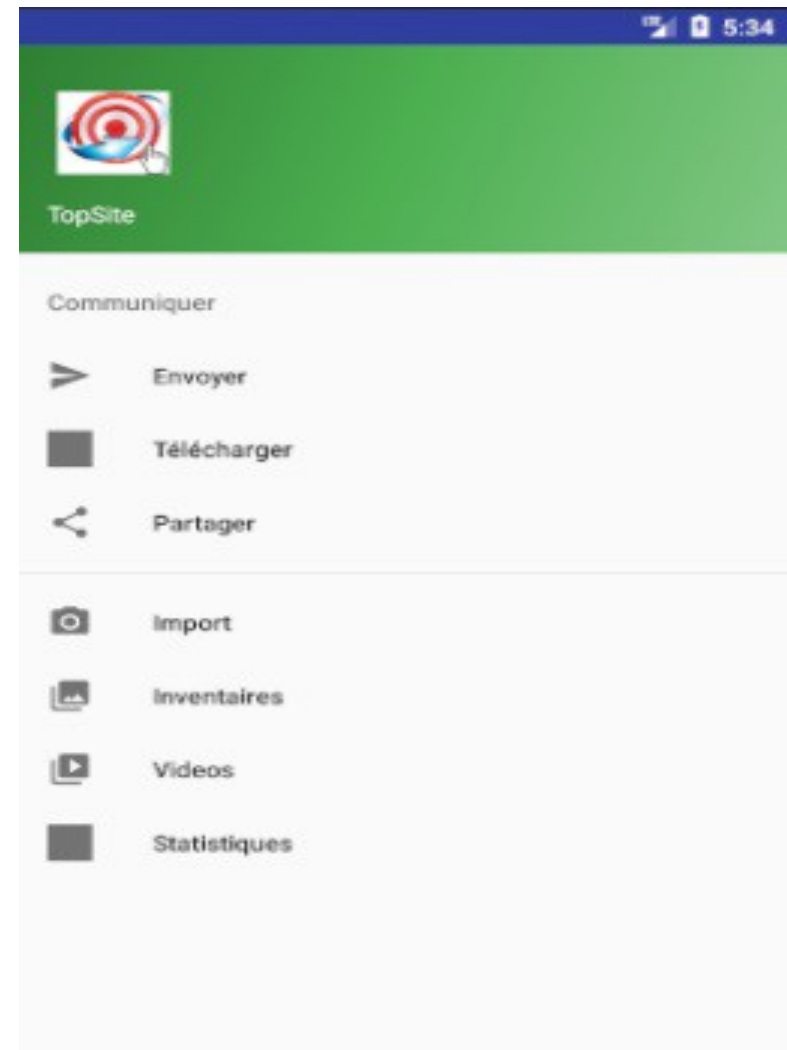
Présentation de l'application TopSite!

- ▶ Réponse du serveur(Utilisateur identifié sur l'application mobile + Token mobile enregistré dans la base de donnée gcm_ids afin d'être en mesure d'envoyer des notifications push sur l'application mobile!)



Présentation de l'application TopSite!

- ▶ Menu principale de l'application TopSite
- ▶ Gagner des crédits: Envoyer et partager sur Facebook, Télécharger!
- ▶ Inventaires de vos documents upload sur votre compte TopSite[Pdf, Mp3, Zip]!
- ▶ Videos You Tube ajouter sur votre compte TopSite!
- ▶ Statistiques!

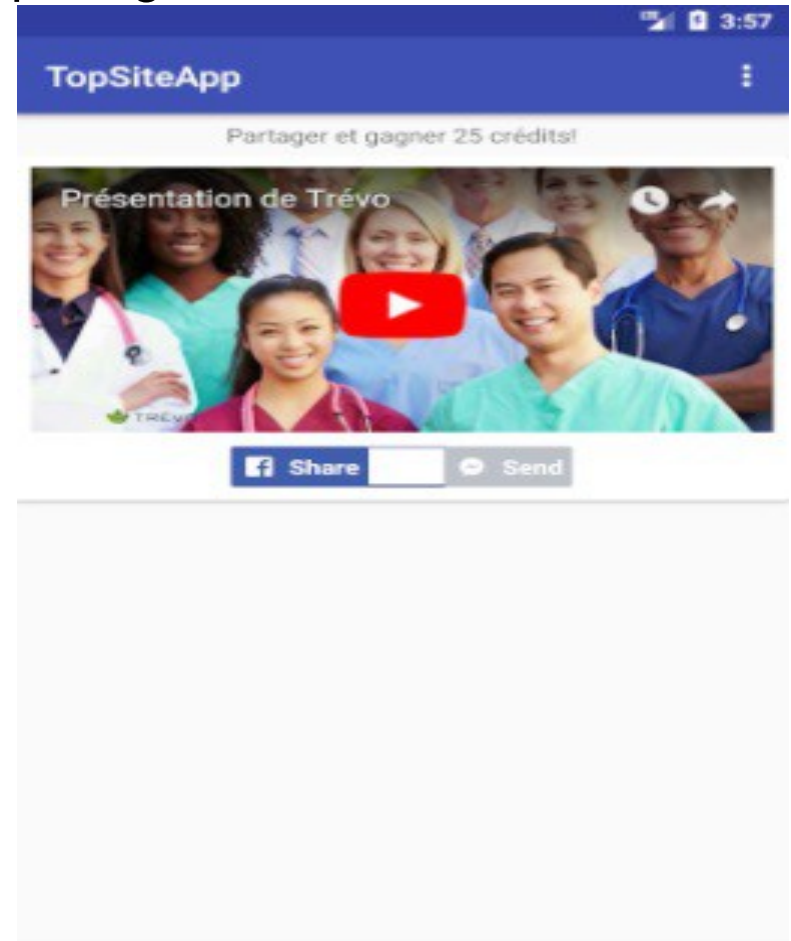


Présentation de l'application TopSite!

- Liste de vos vidéos upload sur votre compte TopSite!



- Partager et gagner 25 crédits par partage!

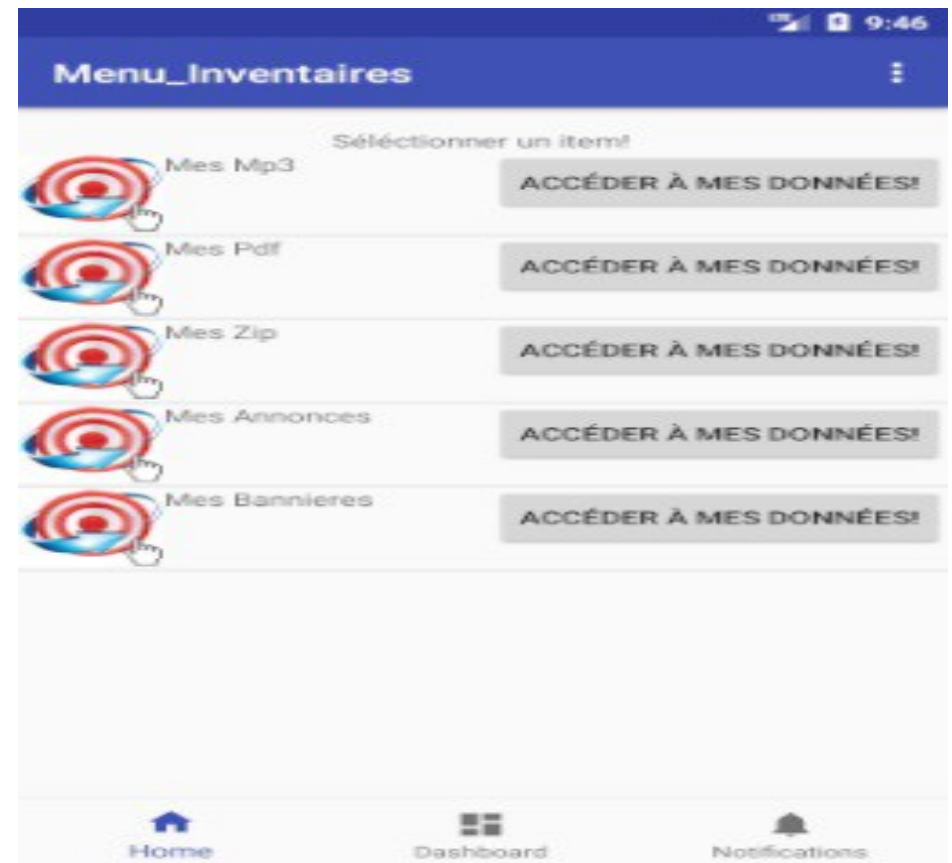


Présentation de l'application TopSite!

- ▶ Gagner des crédits et ajouter vos crédits pour recevoir des téléchargements/partages sur vos fichiers/sites web et recevoir également des notifications push!

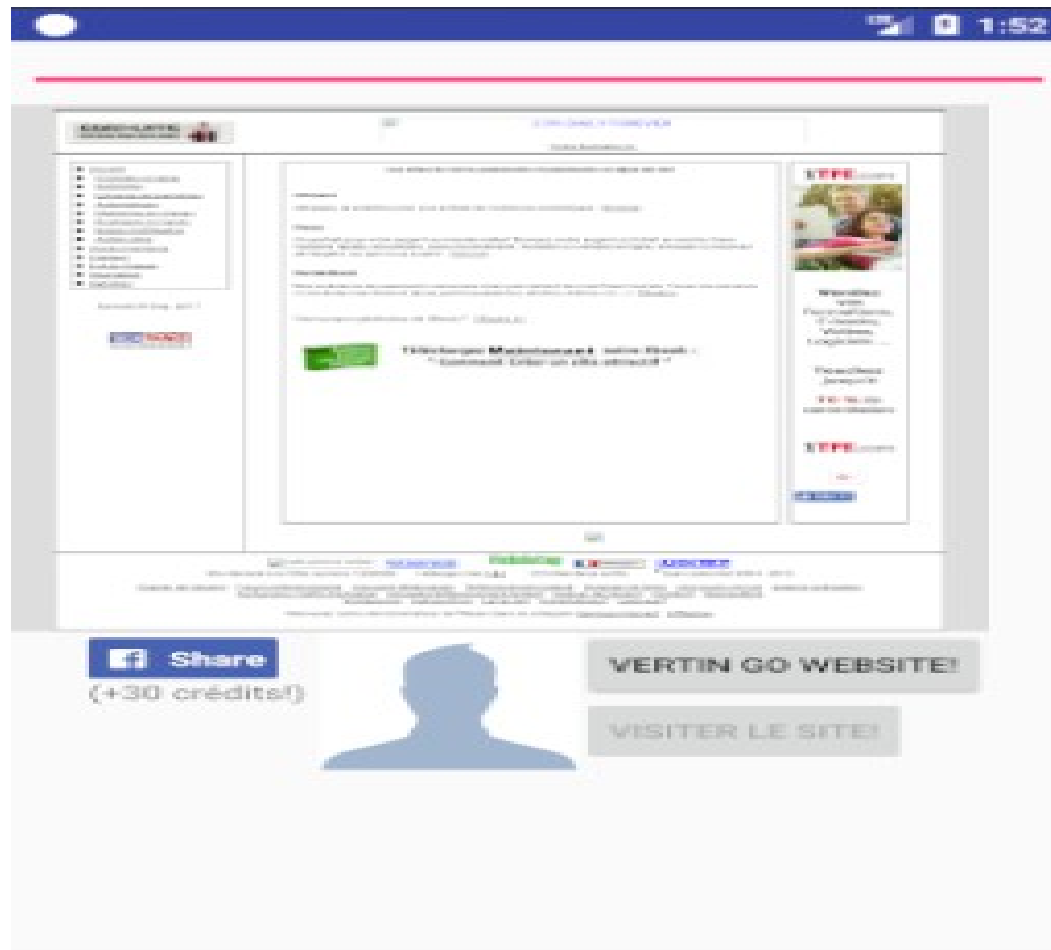


- ▶ Uploader vos fichiers sur votre compte en ligne et récupéré les sur l'app mobile!



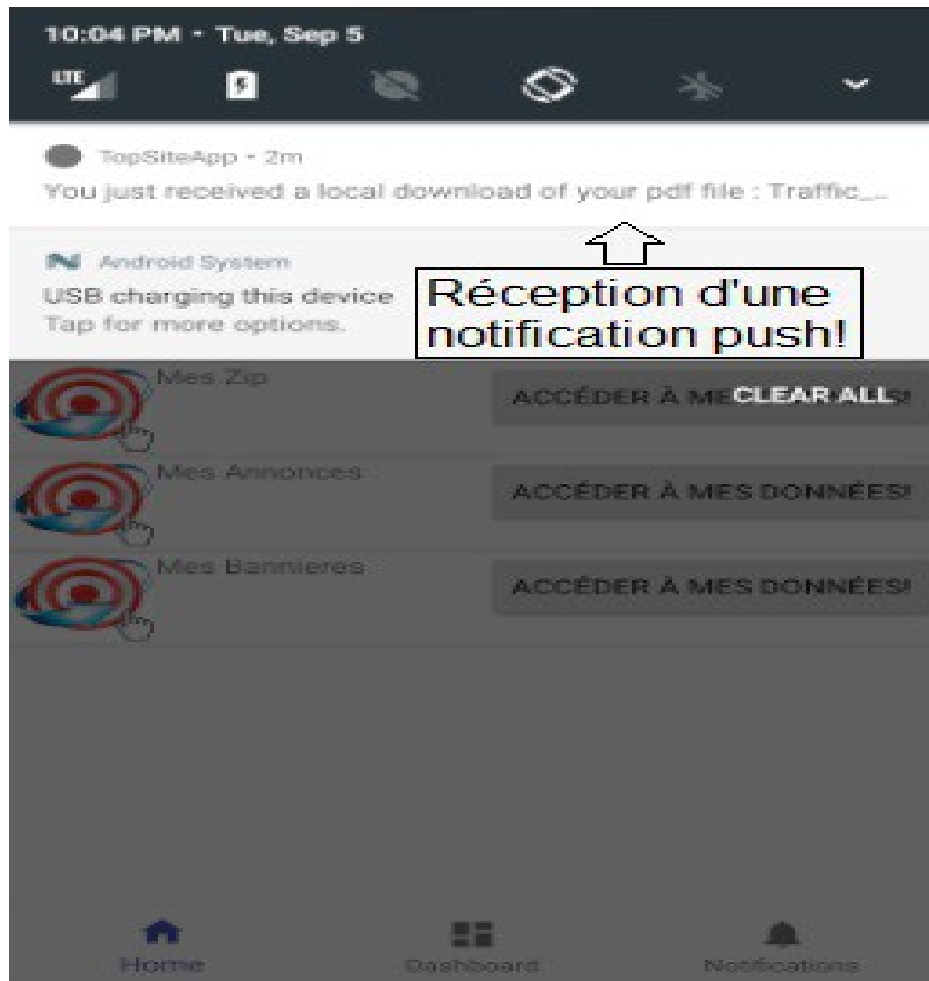
Présentation de l'application TopSite!

- Exemple de l'activité partage de sites web!

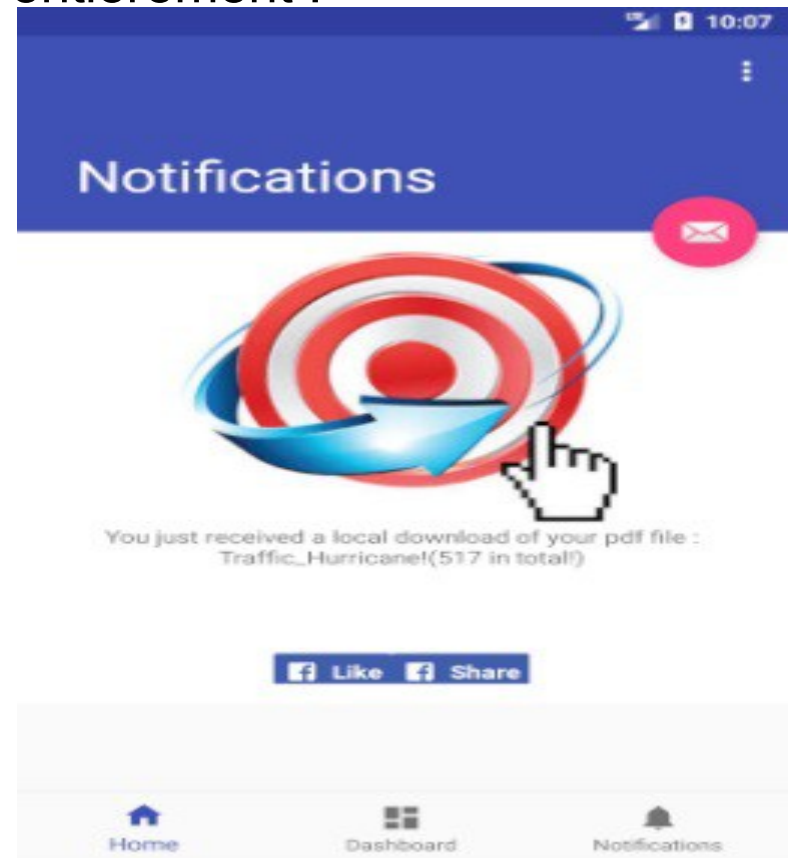


Présentation de l'application TopSite!

- Réception d'une notification push !

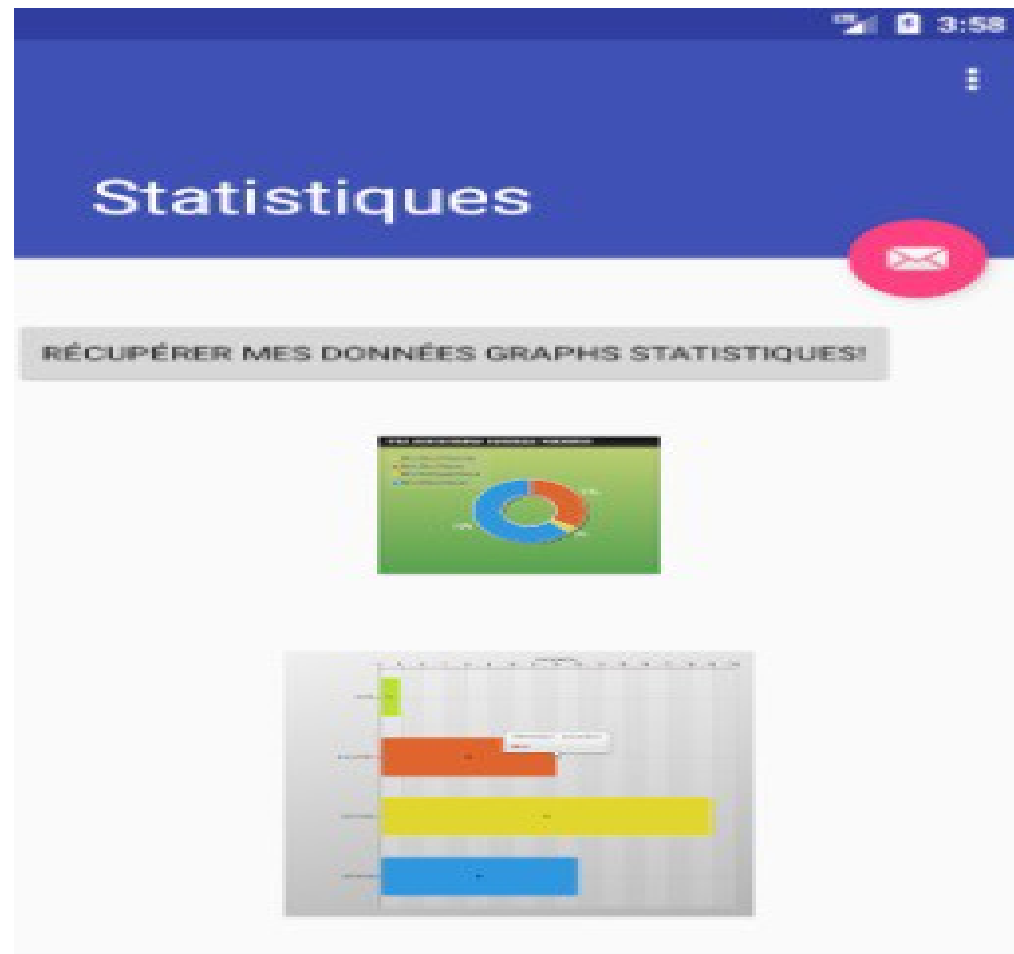


- Suite au clique sur la notification affichage de la notification entièrement !



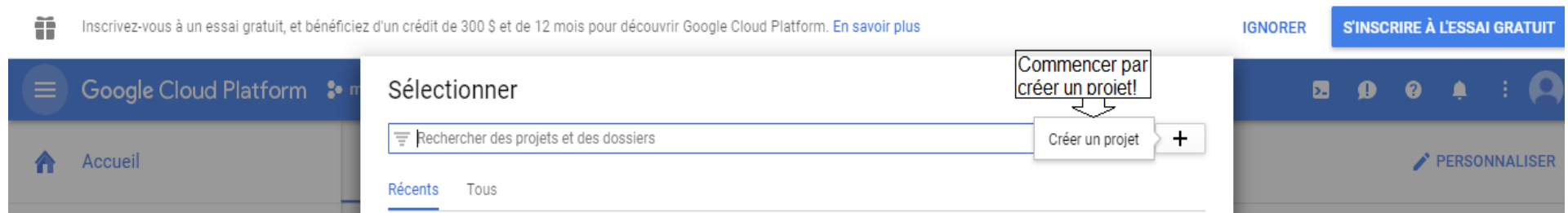
Présentation de l'application TopSite!

► Statistiques

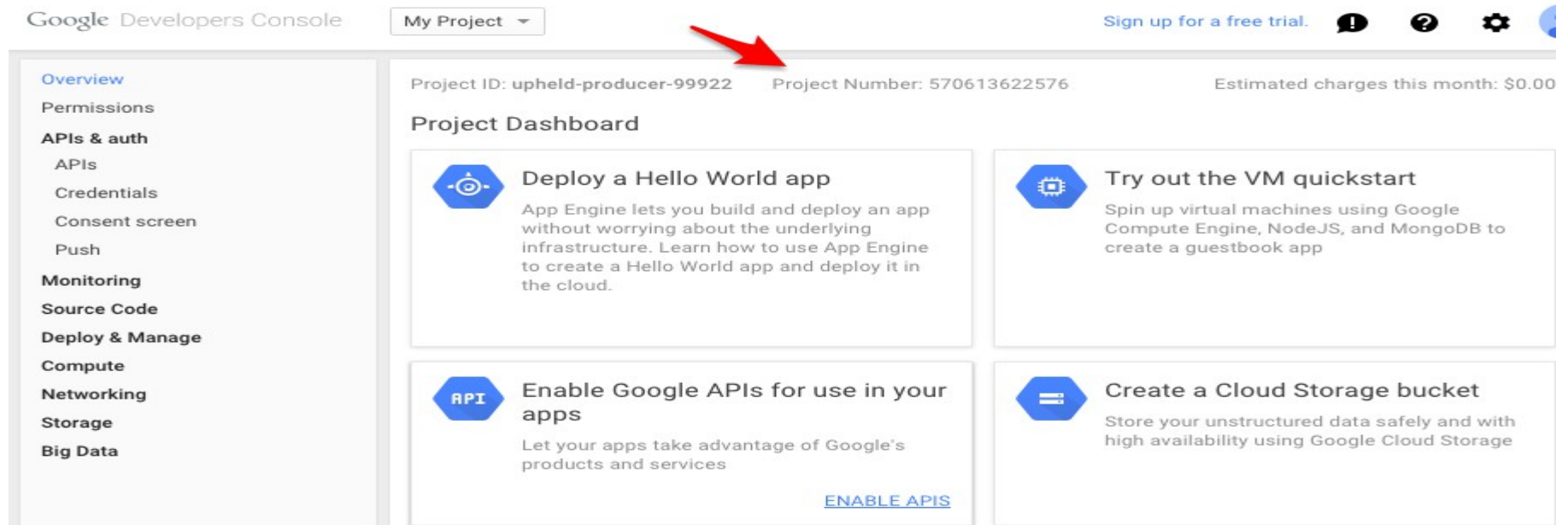


Système de notification push!

► Les étapes dans Google Cloud Plate-Forme



► Noter bien le numéro du projet (Il nous sera utile plus tard)







Système de notification push!

► Rechercher dans API & auth Google Cloud Messaging for Android !


Google Developers Console

My Project ▼

[Sign up for a free trial.](#)    

Overview

Permissions

APIs & auth 

APIs

Credentials

Consent screen

Push

Monitoring

Source Code

Deploy & Manage


Compute

Networking

API Library Enabled APIs (6)

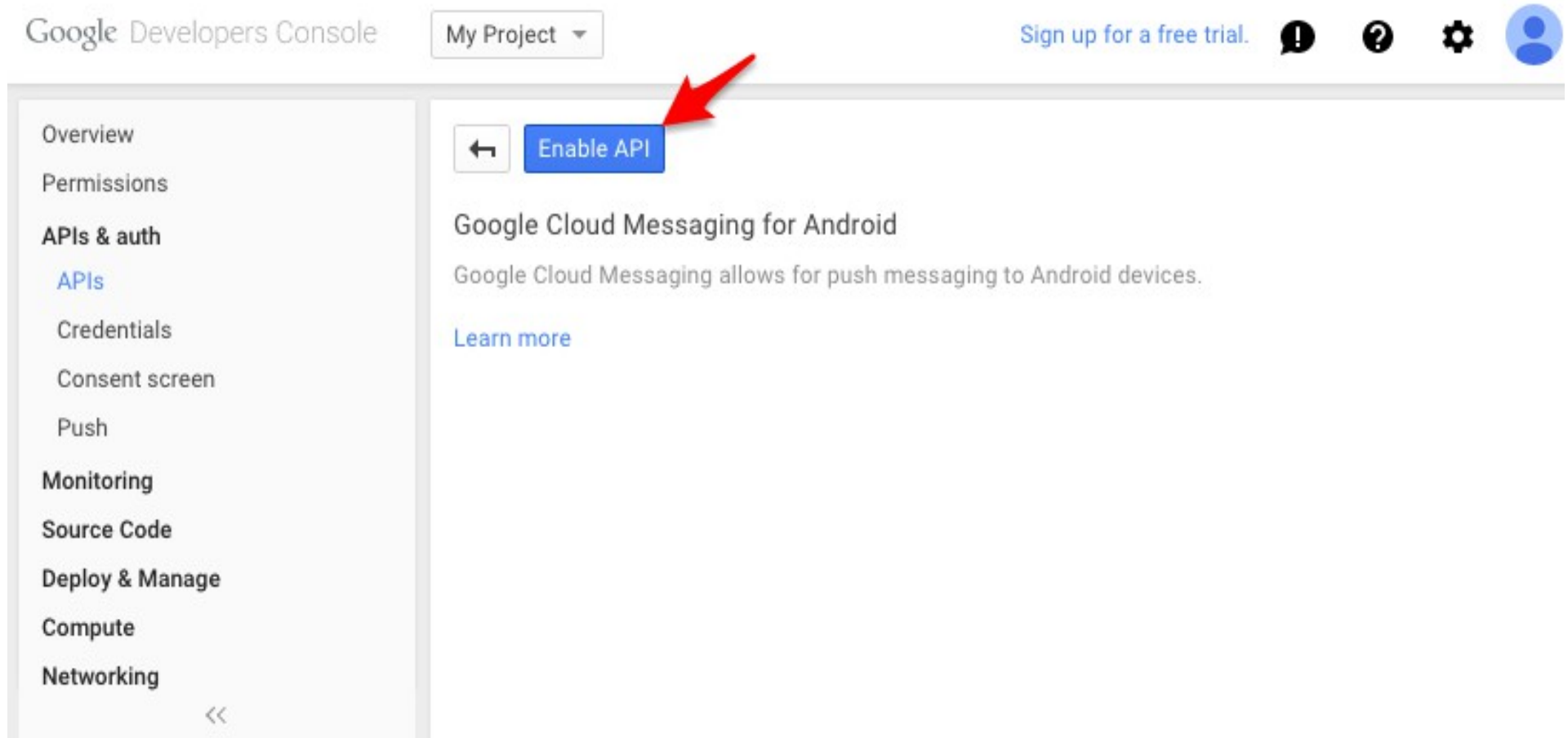
[Back to popular APIs](#)

Name	Description
Google Cloud Messaging for Chrome	Google Cloud Messaging for Chrome allows messages to be delivered from the cloud to apps and extensions running in Chrome.
Google Cloud Messaging for Android	Google Cloud Messaging allows for push messaging to Android devices.
Google Cloud Pub/Sub	Provides reliable, many-to-many, asynchronous messaging between applications.



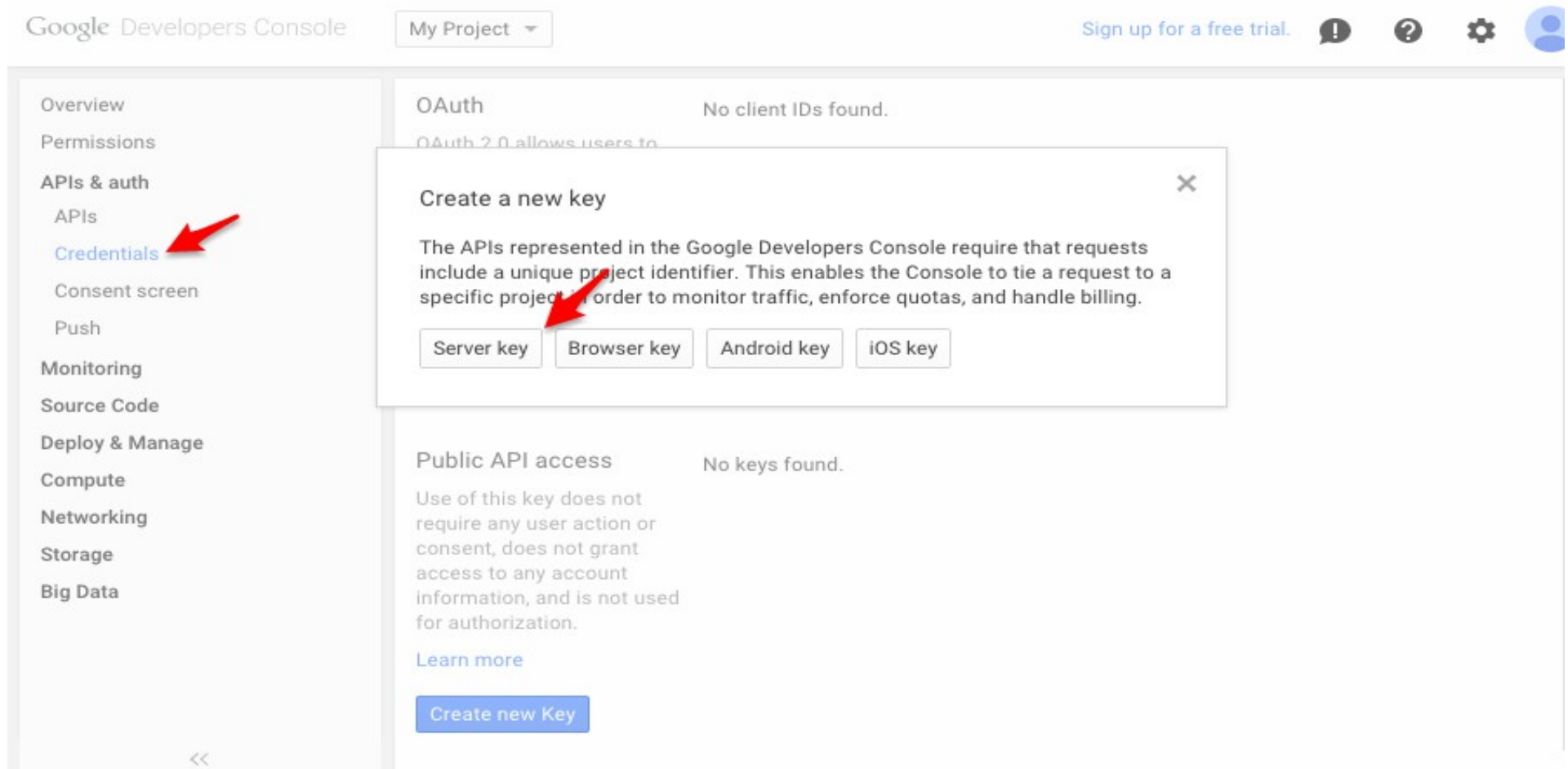
Système de notification push!

- ▶ Activer l'API Google Cloud Messaging for Android!



Système de notification push!

► Créer une clé de serveur!



The screenshot displays the Google Developers Console interface. On the left sidebar, the 'Credentials' link is highlighted with a red arrow. The main content area shows the 'OAuth' section with the message 'No client IDs found.' Below this, a modal dialog titled 'Create a new key' is open. The dialog contains the text: 'The APIs represented in the Google Developers Console require that requests include a unique project identifier. This enables the Console to tie a request to a specific project in order to monitor traffic, enforce quotas, and handle billing.' Below the text are four buttons: 'Server key', 'Browser key', 'Android key', and 'iOS key'. A red arrow points to the 'Server key' button. At the bottom of the dialog, there is a blue button labeled 'Create new Key'.

Google Developers Console My Project Sign up for a free trial. ! ? ⚙️ 👤

Overview
Permissions
APIs & auth
APIs
Credentials
Consent screen
Push
Monitoring
Source Code
Deploy & Manage
Compute
Networking
Storage
Big Data

OAuth No client IDs found.
OAuth 2.0 allows users to...

Create a new key

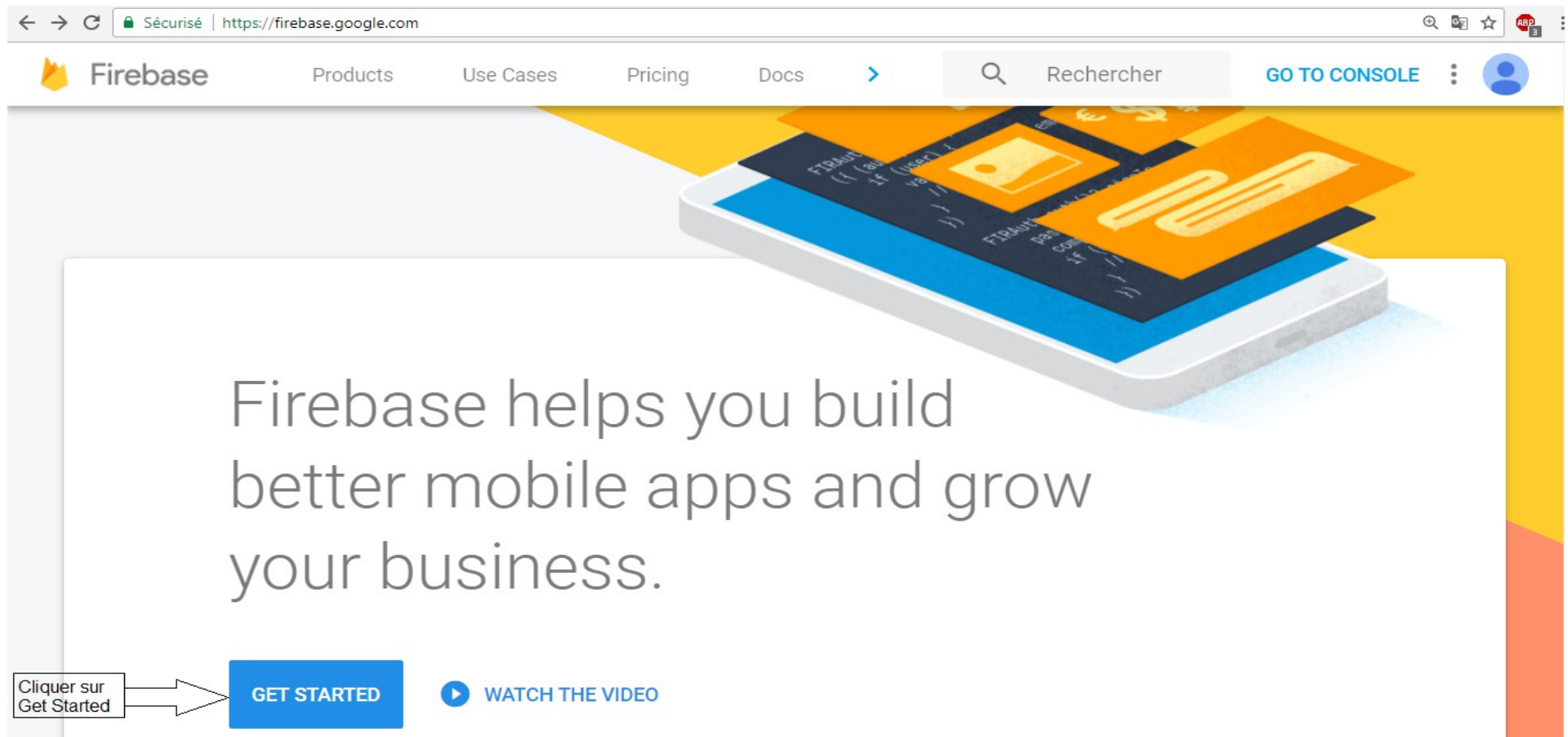
The APIs represented in the Google Developers Console require that requests include a unique project identifier. This enables the Console to tie a request to a specific project in order to monitor traffic, enforce quotas, and handle billing.

Server key Browser key Android key iOS key

Public API access No keys found.
Use of this key does not require any user action or consent, does not grant access to any account information, and is not used for authorization.
[Learn more](#)
Create new Key

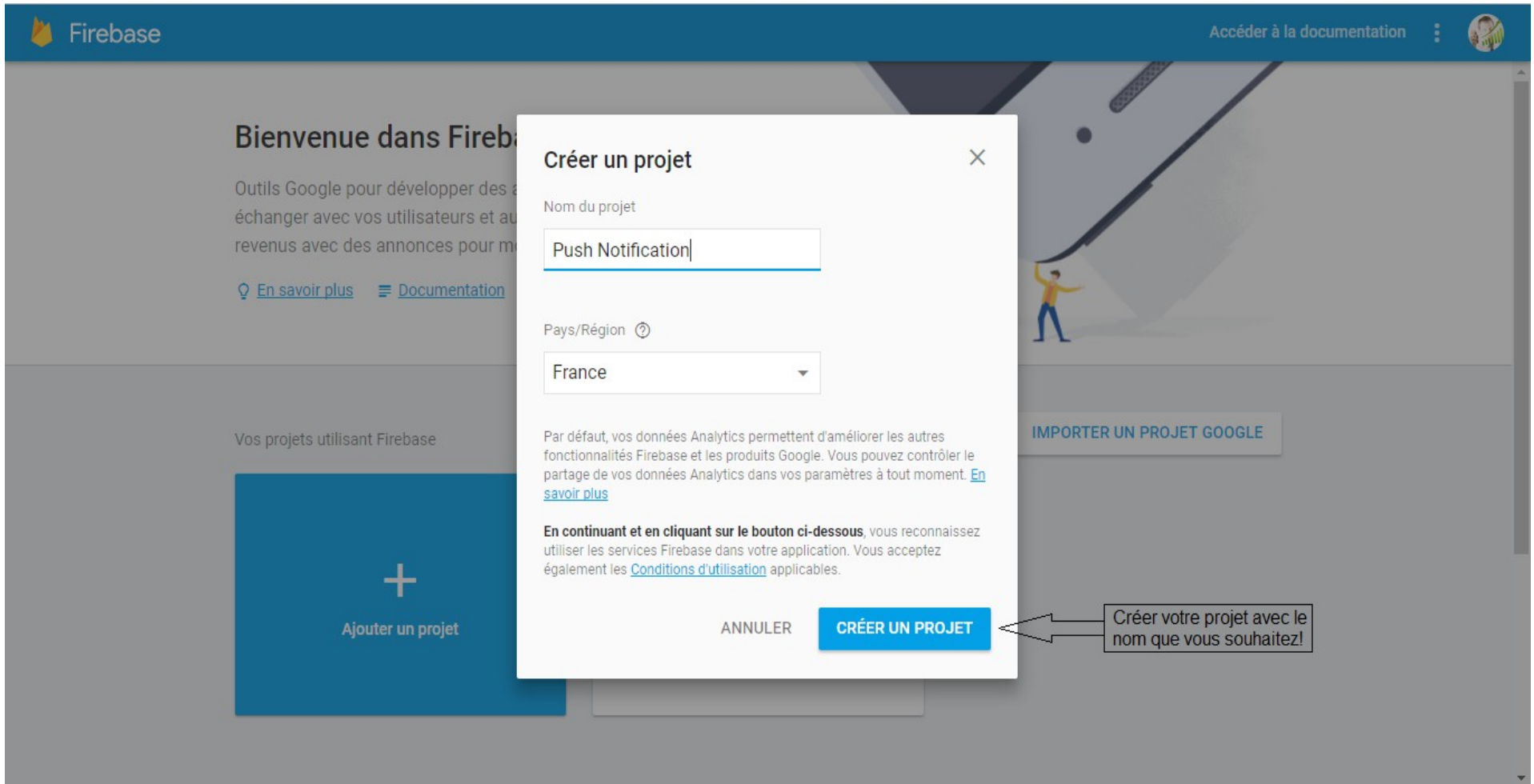
Système de notification push!

► Aller sur: <https://firebase.google.com/>



Système de notification push!

► Créer un projet



The screenshot shows the Firebase console interface. At the top, there's a blue header with the Firebase logo and the text 'Accéder à la documentation'. Below the header, the main content area is titled 'Bienvenue dans Firebase' and includes a large blue button labeled 'Ajouter un projet'. A modal dialog box titled 'Créer un projet' is open in the center. It contains a text input field for 'Nom du projet' with the value 'Push Notification', a dropdown menu for 'Pays/Région' set to 'France', and a blue button labeled 'CRÉER UN PROJET'. A callout box with an arrow points to this button, containing the text 'Créer votre projet avec le nom que vous souhaitez!'. The background is slightly dimmed to show the dialog box clearly.

Créer un projet

Nom du projet

Push Notification

Pays/Région ?

France

Par défaut, vos données Analytics permettent d'améliorer les autres fonctionnalités Firebase et les produits Google. Vous pouvez contrôler le partage de vos données Analytics dans vos paramètres à tout moment. [En savoir plus](#)

En continuant et en cliquant sur le bouton ci-dessous, vous reconnaissez utiliser les services Firebase dans votre application. Vous acceptez également les [Conditions d'utilisation](#) applicables.

ANNULER CRÉER UN PROJET

Créer votre projet avec le nom que vous souhaitez!

Système de notification push!

► Ajouter FireBase à votre application Android!

The screenshot shows the Firebase console interface. The top navigation bar includes the Firebase logo, a 'Push Notification' dropdown, and a link to 'Accéder à la documentation'. The left sidebar lists various services: Overview (selected), Analytics, Authentication, Database, Storage, Hosting, Functions, Test Lab, Crash Reporting, and Performance. The main content area is titled 'Présentation' and displays the text 'Bienvenue dans Firebase. Lancez-vous ici.' above three large circular buttons. The buttons are labeled 'Ajouter Firebase à votre application iOS', 'Ajouter Firebase à votre application Android', and 'Ajouter Firebase à votre application Web'. A callout box with a downward arrow points to the Android button, containing the text 'Ajouter FireBase à votre application Android!'.

Systeme de notification push!

- Enregistré votre application!

Ajouter Firebase à votre application Android

1

2

3

Enregistrer l'application

Télécharger le fichier de configuration

Ajouter le SDK Firebase

Nom du package Android ⓘ

com.myapp|

Pseudo de l'application (facultatif) ⓘ

Application Android Freemium

Certificat de signature de débogage SHA-1 (facultatif) ⓘ

00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00

Requis pour l'assistance relative à Dynamic Links, Invites et Google Sign-In ou l'assistance par téléphone dans Auth. Modifiez les certificats SHA-1 dans les paramètres.

ANNULER

ENREGISTRER L'APPLICATION

dans le projet Push Notification

Indiqué le nom de package de votre projet!

Systeme de notification push!

- Récupérer votre fichier google-service.json

Ajouter Firebase à votre application Android

1

2

3


Enregistrer l'application


Télécharger le fichier de configuration

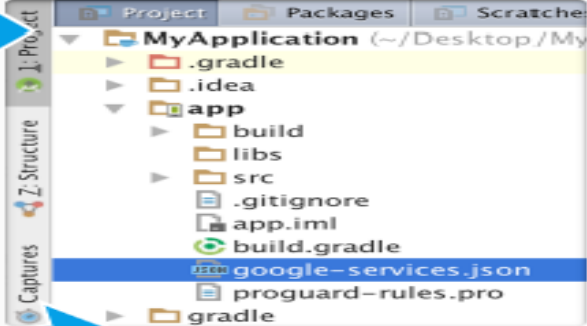
Ajouter le SDK Firebase

Instructions relatives à Android Studio

Alternatives : [Unity](#) [C++](#)

-  **Télécharger google-services.json**
- Accédez à la page **Projet** dans Android Studio pour consulter le répertoire racine de votre projet.
- Déplacez le fichier **google-services.json** que vous venez de télécharger dans le répertoire racine du module de votre application Android.


google-services.json



Récupérer le fichier **google-service.json** et placer le dans le app de votre projet Android!

Système de notification push!

- Ajouter les dépendances à votre projet !

Ajouter Firebase à votre application Android

1

2

3

Enregistrer l'application

Télécharger le fichier de configuration

Ajouter le SDK Firebase

Instructions relatives à Gradle

Alternatives : [Unity](#) [C++](#)

Le plug-in de services Google pour [Gradle](#) ☒ charge le fichier `google-services.json` que vous venez de télécharger. Modifiez vos fichiers `build.gradle` pour utiliser le plug-in.

- Fichier `build.gradle` au niveau du projet (`<project>/build.gradle`):

```
buildscript {
    dependencies {
        // Add this line
        classpath 'com.google.gms:google-services:3.1.0'
    }
}
```
- Fichier `build.gradle` au niveau de l'application (`<project>/<app-module>/build.gradle`):

```
...
// Add to the bottom of the file
apply plugin: 'com.google.gms.google-services'
```

Analytics inclus par défaut ⓘ
- Enfin, appuyez sur **Synchroniser** dans la barre qui apparaît dans l'environnement de développement intégré :

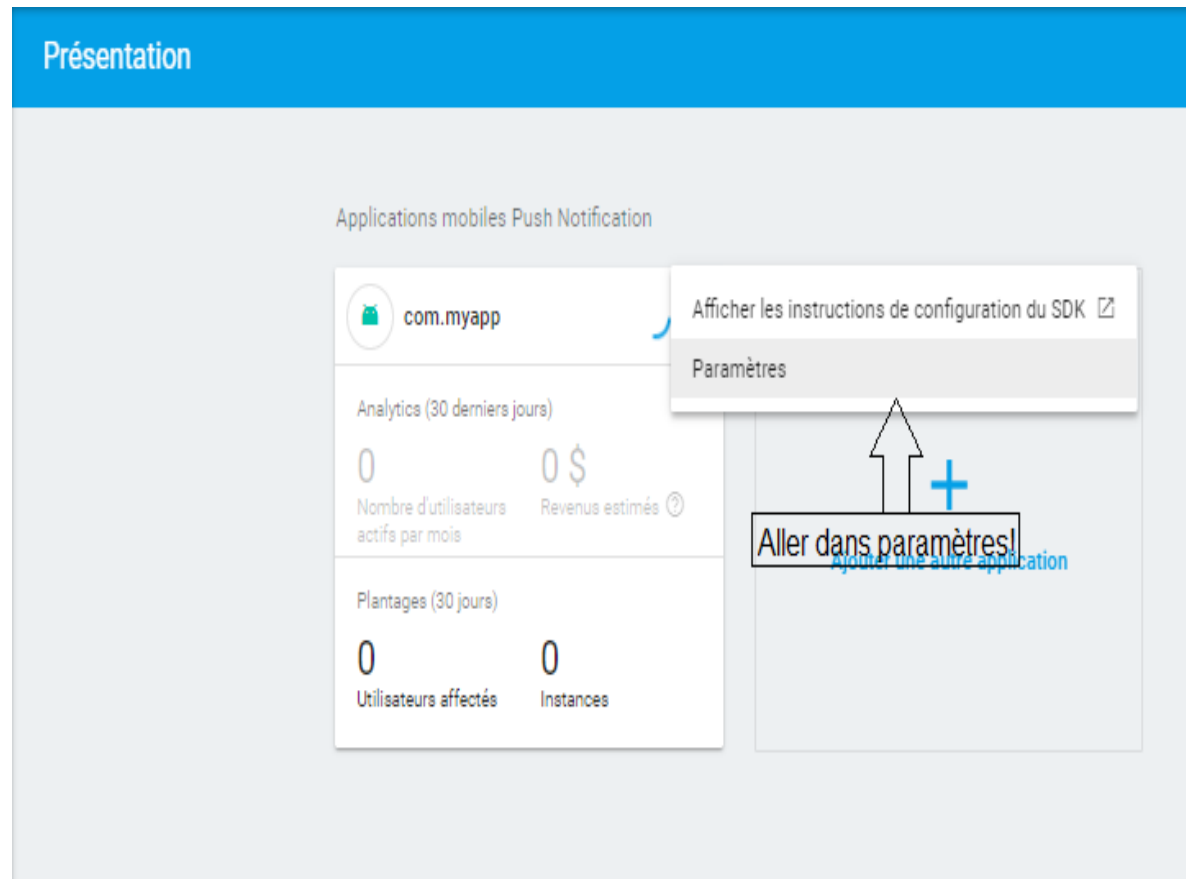
Gradle files have changed since last sync

Sync now

Ajouter les différentes dépendances dans votre projet build.gradle!

Systeme de notification push!

- Récupérer votre clé API Google Cloud Messaging



Systeme de notification push!

- Récupérer votre clé API Google Cloud Messaging !

The screenshot shows the Google Cloud Platform 'Paramètres' (Settings) page for a project. The 'CLOUD MESSAGING' tab is selected, highlighted by an arrow and a text box that says 'Aller au menu Google Cloud Messaging!'. The page displays the following information:

Votre projet	
Nom du projet	Push Notification
Nom public	Push Notification
ID du projet	push-notification-53a00
Clé API Web	Aucune clé API Web pour ce projet

Systeme de notification push!

- Récupérer votre clé d'API Google Cloud Messaging !

Paramètres

GÉNÉRAL CLOUD MESSAGING ANALYTICS ASSOCIATION DE COMPTES COMPTES DE SERVICE

Identifiants du projet

AJOUTER UNE CLÉ DE SERVEUR

Clé	Jeton
Clé de serveur	AAAAcFuxZ-c:APA91bHdMJ0aRH1MmP-TKsAHGdZuokA8ZBQZ0YVxfKyq9DGFLXvrL7oA9LWQd8_CUfDuZPaVU2XvvD-YXCldl0KI13Nhqgt5mrlmOnNfuR1_W0Rh9rBcigydbzOZB8GvEIUqmYt16xT-
Ancienne clé de serveur ...	AlzaSyDz99I-m4VDKvM3P9l8wo5QieBxzwlu1hQ
ID de l'expéditeur ⓘ	
482574690279	

Noter bien cette clé elle nous permettra d'envoyer des notifications push du côté serveur(Web) vers le côté client(App Mobile)

Système de notification push!

► Mise en place du code côté serveur!(Serveur Web)

```
function send_notification($tokens, $message)
{
    $url = 'https://gcm-http.googleapis.com/gcm/send';
    $fields = array(
        'registration_ids' => $tokens,
        'data' => $message
    );

    $headers = array(
        'Authorization: key = votre_clé_d\'API Google.Firebase(GCM)',
        'Content-Type: application/json'
    );

    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL, $url);
    curl_setopt($ch, CURLOPT_POST, true);
    curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 0);
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
    curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($fields));
    $result = curl_exec($ch);
    if ($result === FALSE) {
        die('Curl failed: ' . curl_error($ch));
    }
    curl_close($ch);
    return $result;
}
```

Système de notification push!

- ▶ Mise en place du code côté serveur + création de la base de donnée !
- ▶ Script SQL de la création de la table GCM qui nous permettra entre autre de stocker le token client (L'identifiant qui nous permet d'acheminer un message à une application mobile!)

```
CREATE TABLE IF NOT EXISTS `gcm_ids` (  
    `id` int(11) NOT NULL AUTO_INCREMENT,  
    `gcm_token` text NOT NULL,  
    `date_creation` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE  
CURRENT_TIMESTAMP,  
    PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
```

- ▶ Fonction de récupération du Token Client!

```
function getTokenMobile($id) //Valide  
{  
    $db = new PDO('mysql:host=db_host;dbname=db_name', 'db_user', 'db_pass');  
    $reponse = $db->prepare("select * from gcm_users where id=:id");  
    $reponse->bindValue(':id', $id, PDO::PARAM_INT);  
    $reponse->execute();  
    $row=$reponse->fetch();  
  
    $reponse->closeCursor();  
  
    return $row['token'];  
}
```

Système de notification push!

► Exemple d'envoi d'un message!

```
$tokens = array();  
$token=getTokenMobile(1);  
$tokens[]=$token;  
$message = array("message" => "Salut je suis une notification push!");  
$message_status = send_notification($tokens, $message);  
echo $message_status;
```

Système de notification push!

► Mise en place du code Côté client!

Ajouter les dépendances suivantes dans le fichier
build.gradle (Module App)

```
compile 'com.google.android.gms:play-services-gcm:7.5.+'  
compile 'com.squareup.okhttp:okhttp:2.4.0'
```

Ajouter les ressources string suivantes dans le
fichier strings.xml

```
<string name="gcm_send_message">Identification envoyé au serveur... [Récupérations de vos  
documents...]\n[Récupérations de vos données statistiques...]\n[Activation des notifications push  
mobile...]!</string>  
    <string name="token_error_message">Une erreur s'est produite, veuillez réessayer!</string>  
    <string name="registering_message">Génération d'un token! \n+\n Connexion au serveur Vertin  
Go Website...!</string>
```

Système de notification push!

- Mise en place du code côté client !

Créer un layout activity_main

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:background="@android:color/white" android:orientation="vertical"
    tools:context=".MainActivity">

    <TextView android:text="@string/registering_message"
        android:layout_width="wrap_content" android:layout_height="wrap_content"
        android:id="@+id/informationTextView" android:textAppearance="?
        android:attr/textAppearanceMedium" />

    <ProgressBar android:layout_width="wrap_content" android:layout_height="wrap_content"
        android:id="@+id/registrationProgressBar" />

</LinearLayout>
```

Système de notification push!

► Mise en place du code côté client !(Activité service MyGcmListener)

```
public class MyGcmListenerService extends GcmListenerService {

    private static final String TAG = "MyGcmListenerService";
    /* @param from SenderID of the sender.
     * @param data Data bundle containing message data as key/value pairs.
     * For Set of keys use data.keySet().
     */
    @Override
    public void onMessageReceived(String from, Bundle data) {

        String message = data.getString("message");
        String title = data.getString("title");

        sendNotification(message, title);

    }

    /**
     * Create and show a simple notification containing the received GCM message.
     *
     * @param message GCM message received.
     */
    private void sendNotification(String message, String title) {
        Intent intent = new Intent(this, MainActivity.class);
        intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0 /* Request code */, intent,
            PendingIntent.FLAG_ONE_SHOT);

        Uri defaultSoundUri= RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
        NotificationCompat.Builder notificationBuilder = new NotificationCompat.Builder(this)
            .setSmallIcon(R.mipmap.ic_launcher)
            .setContentTitle(title)
            .setContentText(message)
            .setAutoCancel(true)
            .setSound(defaultSoundUri)
            .setContentIntent(pendingIntent);

        NotificationManager notificationManager =
            (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
        // On génère un nombre aléatoire pour pouvoir afficher plusieurs notifications
        notificationManager.notify(new Random().nextInt(9999), notificationBuilder.build());
    }
}
```

Système de notification push!

- Mise en place du code côté client(Activité service MyInstanceIDListenerService)

```
public class MyInstanceIDListenerService extends InstanceIDListenerService {  
  
    private static final String TAG = "MyInstanceIDLS";  
  
    /**  
     * Called if InstanceID token is updated. This may occur if the security of  
     * the previous token had been compromised. This call is initiated by the  
     * InstanceID provider.  
     */  
    @Override  
    public void onTokenRefresh() {  
        // Fetch updated Instance ID token and notify our app's server of any changes (if  
        applicable).  
        Intent intent = new Intent(this, RegistrationIntentService.class);  
        startService(intent);  
    }  
}
```


Système de notification push!

► Mise en place du code côté client(Class QuickStartPreferences)

```
public class QuickstartPreferences {  
  
    public static final String SENT_TOKEN_TO_SERVER = "sentTokenToServer";  
    public static final String REGISTRATION_COMPLETE = "registrationComplete";  
  
}
```

Système de notification push!

- Mise en place du code côté client(Activité service RegistrationIntentService)

```
public class RegistrationIntentService extends IntentService {

    private static final String TAG = "RegIntentService";

    private static final String REGISTER_URL = "http://192.168.1.9/gcm/register.php";

    private static final String KEY_TOKEN = "gcm_token";

    public RegistrationIntentService() {
        super(TAG);
    }

    @Override
    protected void onHandleIntent(Intent intent) {
        SharedPreferences sharedPreferences = PreferenceManager.getDefaultSharedPreferences(this);

        try {
            // In the (unlikely) event that multiple refresh operations occur simultaneously,
            // ensure that they are processed sequentially.
            synchronized (TAG) {
                // [START register_for_gcm]
                // Initially this call goes out to the network to retrieve the token, subsequent calls
                // are local.
                InstanceID instanceID = InstanceID.getInstance(this);

                String token = instanceID.getToken(getString(R.string.gcm_defaultSenderId),
                    GoogleCloudMessaging.INSTANCE_ID_SCOPE, null);

                Log.i(TAG, "GCM Registration Token: " + token);

                // Si le token a déjà été enregistré pas la peine de le renvoyer
                if (!sharedPreferences.getBoolean(QuickstartPreferences.SENT_TOKEN_TO_SERVER, false))
                    sendRegistrationToServer(token);

                // You should store a boolean that indicates whether the generated token has been
                // sent to your server. If the boolean is false, send the token to your server,
                // otherwise your server should have already received the token.
                sharedPreferences.edit().putBoolean(QuickstartPreferences.SENT_TOKEN_TO_SERVER, true).apply();
            }
        } catch (Exception e) {
            Log.d(TAG, "Failed to complete token refresh", e);
            // If an exception happens while fetching the new token or updating our registration data
            // on a third-party server, this ensures that we'll attempt the update at a later time.
            sharedPreferences.edit().putBoolean(QuickstartPreferences.SENT_TOKEN_TO_SERVER, false).apply();
        }

        // Notify UI that registration has completed, so the progress indicator can be hidden.
        Intent registrationComplete = new Intent(QuickstartPreferences.REGISTRATION_COMPLETE);
        LocalBroadcastManager.getInstance(this).sendBroadcast(registrationComplete);
    }
}
```

Système de notification push!

- Mise en place du code côté client!(Fonction à rajouté à la classe précédente!)

```
/**
 * Ici nous allons envoyer le token de l'utilisateur au serveur
 *
 * @param token Le token
 */
private void sendRegistrationToServer(String token) {

    OkHttpClient client = new OkHttpClient();

    RequestBody requestBody = new FormEncodingBuilder()
        .add(KEY_TOKEN, token)
        .build();

    Request request = new Request.Builder()
        .url(REGISTER_URL)
        .post(requestBody)
        .build();

    try {
        Response response = client.newCall(request).execute();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

Système de notification push!

► Mise en place du code côté client(Class MainActivity)

```
public class MainActivity extends AppCompatActivity {

    private static final int PLAY_SERVICES_RESOLUTION_REQUEST = 9000;
    private static final String TAG = "MainActivity";

    private BroadcastReceiver mRegistrationBroadcastReceiver;
    private ProgressBar mRegistrationProgressBar;
    private TextView mInformationTextView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mRegistrationProgressBar = (ProgressBar) findViewById(R.id.registrationProgressBar);
        mRegistrationBroadcastReceiver = new BroadcastReceiver() {
            @Override
            public void onReceive(Context context, Intent intent) {
                mRegistrationProgressBar.setVisibility(ProgressBar.GONE);
                SharedPreferences sharedPreferences =
                    PreferenceManager.getDefaultSharedPreferences(context);
                boolean sentToken = sharedPreferences
                    .getBoolean(QuickstartPreferences.SENT_TOKEN_TO_SERVER, false);
                if (sentToken) {
                    mInformationTextView.setText(getString(R.string.gcm_send_message));
                } else {
                    mInformationTextView.setText(getString(R.string.token_error_message));
                }
            }
        };
        mInformationTextView = (TextView) findViewById(R.id.informationTextView);

        if (checkPlayServices()) {
            // Start IntentService to register this application with GCM.
            Intent intent = new Intent(this, RegistrationIntentService.class);
            startService(intent);
        }
    }
}
```

Système de notification push!

- Mise en place du code côté client(Fonction à ajouter dans la classe MainActivity)

```
@Override
protected void onResume() {
    super.onResume();
    LocalBroadcastManager.getInstance(this).registerReceiver(mRegistrationBroadcastReceiver,
        new IntentFilter(QuickstartPreferences.REGISTRATION_COMPLETE));
}

@Override
protected void onPause() {
    LocalBroadcastManager.getInstance(this).unregisterReceiver(mRegistrationBroadcastReceiver);
    super.onPause();
}

/**
 * Vérifier si notre utilisateur a l'application Google Play Service
 */
private boolean checkPlayServices() {
    int resultCode = GooglePlayServicesUtil.isGooglePlayServicesAvailable(this);
    if (resultCode != ConnectionResult.SUCCESS) {
        if (GooglePlayServicesUtil.isUserRecoverableError(resultCode)) {
            GooglePlayServicesUtil.getErrorDialog(resultCode, this,
                PLAY_SERVICES_RESOLUTION_REQUEST).show();
        } else {
            Log.i(TAG, "This device is not supported.");
            finish();
        }
        return false;
    }
    return true;
}
```

Système de notification push!

► Mise en place du code côté client(Fichier AndroidManifest)

```
<?xml version="1.0" encoding="utf-8"?>
  <manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.androidpourtous.gcm" >

    <uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />

    <application android:allowBackup="true" android:icon="@mipmap/ic_launcher" android:label="@string/app_name"
android:theme="@style/AppTheme" >

      <activity android:name="com.androidpourtous.gcm.MainActivity" android:label="@string/app_name" >
        <intent-filter>
          <action android:name="android.intent.action.MAIN" />

          <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
      </activity>

      <receiver android:name="com.google.android.gms.gcm.GcmReceiver" android:exported="true"
android:permission="com.google.android.c2dm.permission.SEND" >
        <intent-filter>
          <action android:name="com.google.android.c2dm.intent.RECEIVE" />
          <category android:name="com.androidpourtous.gcm" />
        </intent-filter>
      </receiver>

      <service android:name="com.androidpourtous.gcm.MyGcmListenerService" android:exported="false" >
        <intent-filter>
          <action android:name="com.google.android.c2dm.intent.RECEIVE" />
        </intent-filter>
      </service>

      <service android:name="com.androidpourtous.gcm.MyInstanceIdListenerService" android:exported="false">
        <intent-filter>
          <action android:name="com.google.android.gms.iid.InstanceID"/>
        </intent-filter>
      </service>

      <service android:name="com.androidpourtous.gcm.RegistrationIntentService" android:exported="false">
      </service>

    </application>
  </manifest>
```

Ajout du SDK Facebook + Bouton de partage!

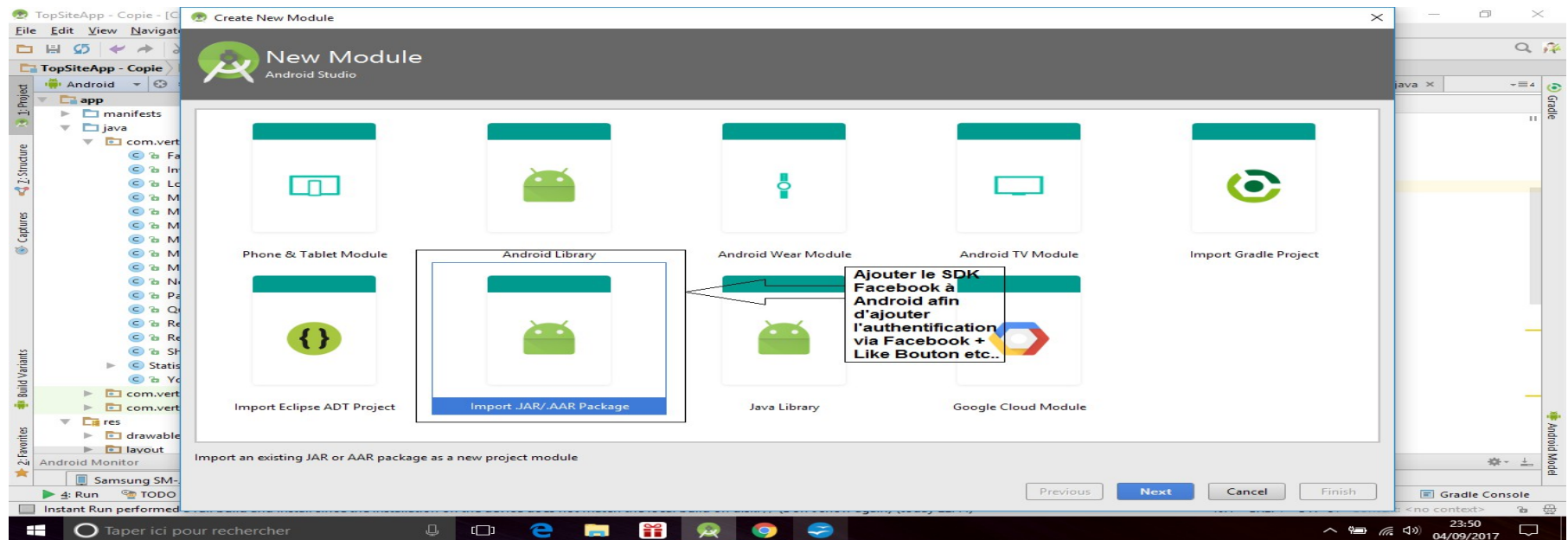
1) Importation du SDK à votre projet Android!

Télécharger le SDK à l'adresse suivante :

https://developers.facebook.com/docs/android/downloads/?locale=fr_FR

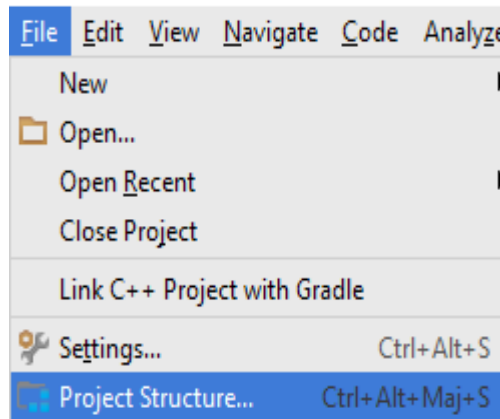
Dézipper le contenu et dans votre projet Android en sélectionnant le dossier app + clic droit dans votre arborescence de votre projet à gauche!

Faites New → Module → Import an existing JAR package → Sélectionner le fichier .JAR du SDK télécharger normalement appelé : facebook-android-sdk-4.26.0

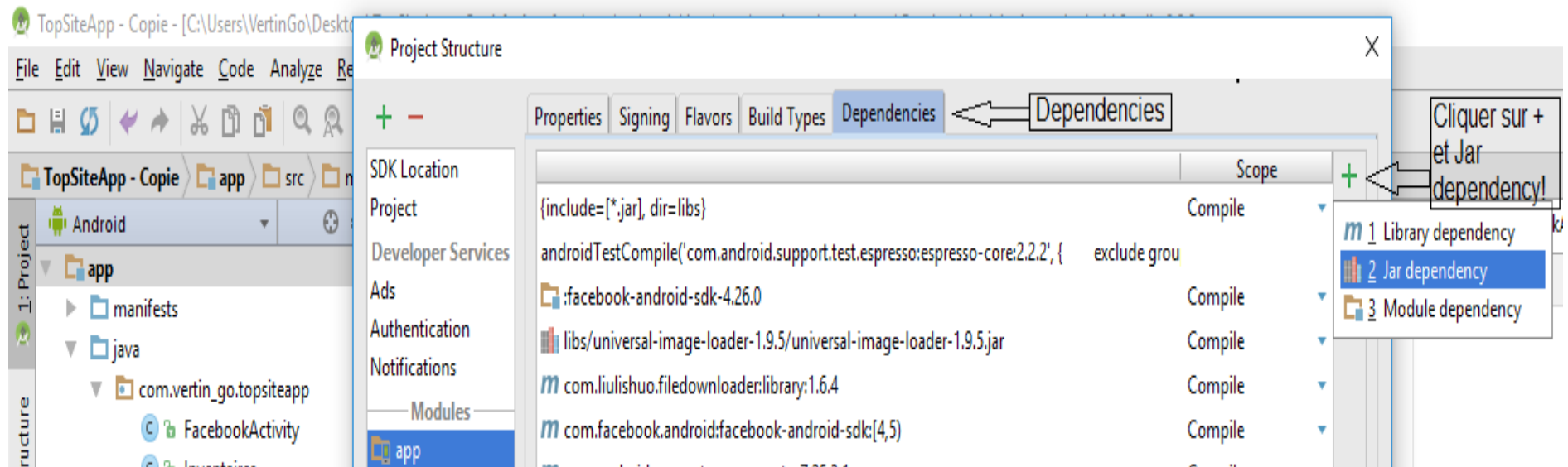


Ajout du SDK Facebook + Bouton de partage!

- ▶ Après l'importation de votre library .JAR aller sur File → Project Structure



- ▶ Et dans Module App Dependencies indiquer les chemins vers le fichier .JAR spécifier à l'étape précédente !



Ajout du SDK Facebook + Bouton de partage!

- ▶ 2) Ajout des dépendances dans votre projet et importation des packages sdk Facebook !

- ▶ Dans votre fichier build.gradle (Module : app)

Ajouter mavenCentral() dans repositories au dessus de dependencies:

```
repositories{  
    mavenCentral()  
}
```

- ▶ Ajouter également dans dependencies:

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

Importer les packages suivant dans votre class ou vous souhaitez configurer une connexion via facebook par exemple:

```
import com.facebook.FacebookSdk;  
import com.facebook.login.LoginResult;
```

Ajout du SDK Facebook + Bouton de partage!

► Ajout dans le fichier Android Manifest:

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

<meta-data
    android:name="com.facebook.sdk.ApplicationId"
    android:value="@string/facebook_app_id" />

<provider
    android:name="com.facebook.FacebookContentProvider"
    android:authorities="com.facebook.app.FacebookContentProvider{fb_app_id}"
    android:exported="true" />

<activity
    android:name="com.facebook.FacebookActivity"
    android:configChanges="keyboard|keyboardHidden|screenLayout|screenSize|orientation"
    android:label="@string/app_name" />
<activity
    android:name="com.facebook.CustomTabActivity"
    android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.VIEW" />
        <category android:name="android.intent.category.DEFAULT" />
        <category android:name="android.intent.category.BROWSABLE" />
    <data android:scheme="@string/fb_login_protocol_scheme" />
    </intent-filter>
</activity>
```

Ajout du SDK Facebook + Bouton de partage!

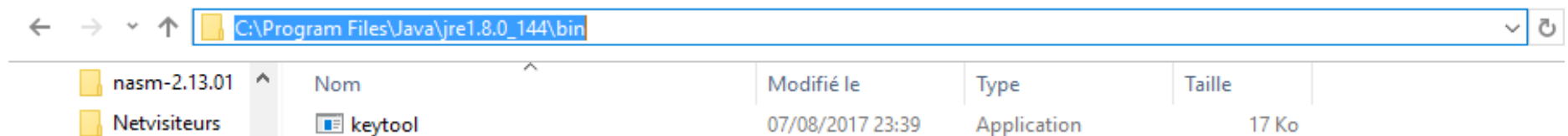
► Ajout dans les ressources strings !

```
<string name="facebook_app_id">votre_app_id</string>  
<string name="fb_login_protocol_scheme">fb|votre_app_id</string>
```

Ajout du SDK Facebook + Bouton de partage!

- ▶ 3) Générez une clé de hachage et renseignez-la sur votre gestionnaire d'application sur votre compte Facebook afin d'établir un lien fiable entre votre application et Facebook!

(Taper cmd dans le champs recherche + Ctrl + Shift + Entrée et placez-vous dans le dossier : **C:\Program Files\Java\jre1.8.0_144\bin** avec l'outil keytool illustrer ci-dessous afin d'exécuter les commandes de génération de clé de hachage!)



Ci-dessous la commande à exécuter dans une commande prompt en tant qu'administrateur! (Si vous n'avez pas OpenSSL télécharger le et veillez à ce que les chemins vers openssl en pipeline correspondent bien à ceux indiqués dans la commande ci-dessous!):

```
keytool -exportcert -alias androiddebugkey -keystore  
chemin_vers_votre_fichier_debug.keystore
```

```
C:\Users\VertinGo\.android\debug.keystore |  
C:\OpenSSL\bin\openssl sha1 -binary | C:\OpenSSL\bin\openssl  
base64
```

Ajout du SDK Facebook + Bouton de partage!

- ▶ On vous demande un mot de passe:

Taper android

```
Administrateur : Invite de commandes
Microsoft Windows [version 10.0.14393]
(c) 2016 Microsoft Corporation. Tous droits réservés.

C:\Windows\system32> cd..

C:\Windows> cd..

C:\> cd C:\Program Files\Java\jre1.8.0_144\bin

C:\Program Files\Java\jre1.8.0_144\bin> keytool -exportcert -alias androiddebugkey -keystore C:\Users\VertinGo\.android\
debug.keystore | C:\OpenSSL\bin\openssl sha1 -binary | C:\OpenSSL\bin\openssl base64
WARNING: can't open config file: C:/OpenSSL/openssl.cnf
WARNING: can't open config file: C:/OpenSSL/openssl.cnf
Entrez le mot de passe du fichier de clés : android
RyC7RYsiRXWDT/lvFtyQ8pNL26E=
```

Et voilà votre clé de hachage à renseigner sur votre gestionnaire d'application sur votre compte Facebook dans paramètres Général, ajouter une plate-forme android si ce n'est pas déjà fait et renseigné les champs comme illustré ci-dessous!

Android

Démarrage rapide

Nom du package Google Play

com.vertin_go.topsiteapp

Le nom de package de votre projet!

Nom de classe

MainActivity

Le nom de la classe principale!

Hachages clés

RyC7RYsiRXWDT/lvFtyQ8pNL26E=

Indiquer votre clé de hachage!

URL de l'App Store Amazon (facultatif)

Ex. : <http://www.amazon.com/dp/B004GJDQT8>

☒ Oui

Authentification unique

Sera lancé à partir des applications Android

☐ Non

Lien profond

Les liens du fil d'actualité lancent cette app

Exemple d'activité Facebook Login!

```
public class FacebookActivity extends AppCompatActivity {

    LoginButton loginButton;
    TextView textview;
    CallbackManager callbackManager;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        FacebookSdk.sdkInitialize(getApplicationContext());
        setContentView(R.layout.activity_facebook);

        loginButton=(LoginButton)findViewById(R.id.login_button);
        loginButton.setReadPermissions("email");
        textview=(TextView)findViewById(R.id.loginstatus);
        callbackManager = CallbackManager.Factory.create();

        loginButton.registerCallback(callbackManager,
            new FacebookCallback<LoginResult>() {
                @Override
                public void onSuccess(LoginResult loginResult) {
                    textview.setText("Connexion réussie \n" + loginResult.getAccessToken().getUserId());
                }

                @Override
                public void onCancel() {
                    textview.setText("Connexion annulé");
                }

                @Override
                public void onError(FacebookException exception) {
                    // App code
                }
            });
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {

        callbackManager.onActivityResult(requestCode, resultCode, data);
    }
}
```

Exemple d'activité Facebook Login

► Ajout du bouton dans le layout!

```
<com.facebook.login.widget.LoginButton  
    android:id="@+id/login_button"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginTop="5dp"  
    android:layout_gravity="center"  
    android:layout_centerInParent='true' />
```

Comment nous suivre?

► You Tube



► FaceBook

