



Android et le cloud

Plan de l'exposé

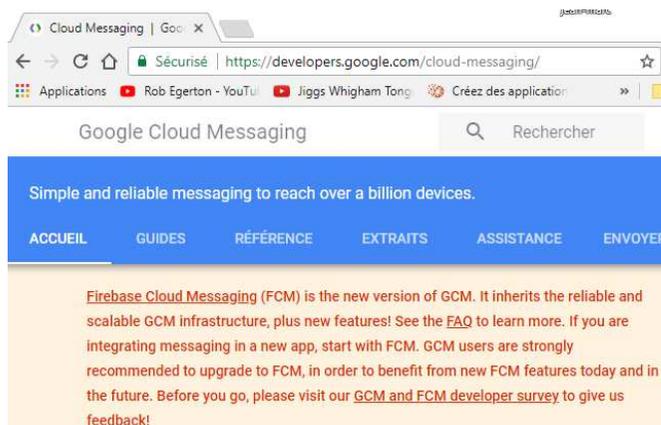


- ❑ But de l'exposé
- ❑ Présentation de Firebase
- ❑ Construction d'un projet dans Firebase
- ❑ L'application distribuée Analytics
- ❑ L'application distribuée Firebase Cloud Messaging

Le but de cet exposé

- ❑ Faire communiquer des smartphones Android avec le cloud
- ❑ Faire communiquer des smartphones Android entre eux grâce au cloud
- ❑ Historique GCM (Google Cloud Messaging)
- ❑ Voir "Lab on Android Platforms, Cloud Computing" école "Secure Smart Object & Internet of Things 23-25 septembre 2014" à <http://cedric.cnam.fr/~farinone/SSOIoT/>

❑ Mais



d'où Firebase Cloud Messaging (FCM)

Présentation de Firebase

- ❑ Firebase = {services d'hébergement pour applications Android, iOS, ...}
- ❑ Hébergement en NoSQL de contenu ou de services (authentification, serveur de communication temps réel)
- ❑ Lancé en 2011 (Envolv) et racheté par Google en octobre 2014
- ❑ source : <https://fr.wikipedia.org/wiki/Firebase>
- ❑ Voir aussi <https://en.wikipedia.org/wiki/Firebase>
- ❑ condition financière d'utilisation à <https://firebase.google.com/pricing/>

Firebase : pour débiter



- ❑ La vidéo qui se trouve à <https://firebase.google.com/docs/android/setup?authuser=0> (Add Firebase to Your Android Project) ou bien <https://www.youtube.com/watch?v=cNPCgJW8c-E> fonctionne
- ❑ 3 étapes :
 - ❑ vérifier les numéros de version de parties d'Android Studio
 - ❑ créer un projet Firebase
 - ❑ écrire une Android app se connectant à ce projet

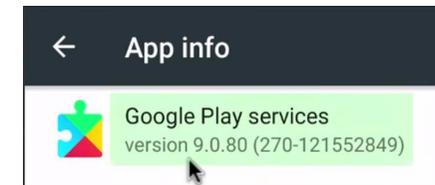
Firebase et Android : des exemples



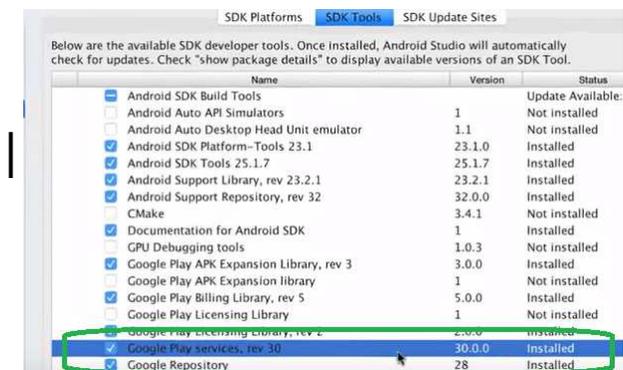
- ❑ Récupérer les exemples de `https://github.com/firebase/quickstart-android`
- ❑ Dézipper
- ❑ Dans Android Studio charger (Close Project puis "Open an existing Android Studio project"), l'app Analytics

Numéros de version de parties d'Android Studio

- ❑ Vérifier que le smartphone et/ou l'AVD ont :
 - ❑ \geq Android 2.3 (API 9) ayant les Google Play Services de version \geq 9.0
 - ❑ Les Google Play Services (Services Google Play en français n'est pas jean-marc) de version \geq 9.0
Pour voir le numéro de version, faire Paramètres | Applications | Services Google Play.
Le numéro de version est dessous du titre :



- ❑ Vérifier qu'Android Studio a installé :
 - ❑ Google Play Services de version \geq 30.
Pour cela SDK Manager | onglet SDK Tools | Google Play Services
 - ❑ Google repository de version \geq 26
- ❑ On est prêt à écrire une Android app utilisant Firebase

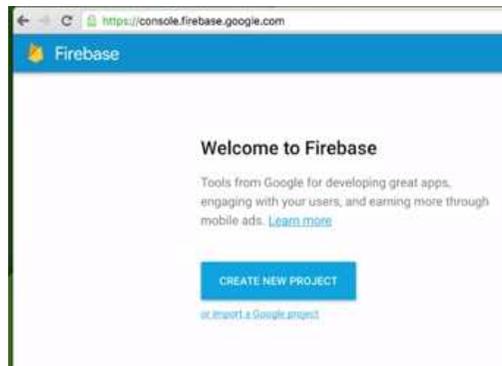


Name	Version	Status
Android SDK Build Tools		Update Available:
Android Auto API Simulators	1	Not installed
Android Auto Desktop Head Unit emulator	1.1	Not installed
Android SDK Platform-Tools 23.1	23.1.0	Installed
Android SDK Tools 25.1.7	25.1.7	Installed
Android Support Library, rev 23.2.1	23.2.1	Installed
Android Support Repository, rev 32	32.0.0	Installed
CMake	3.4.1	Not installed
Documentation for Android SDK	1	Installed
GPU Debugging tools	1.0.3	Not installed
Google Play APK Expansion Library, rev 3	3.0.0	Installed
Google Play APK Expansion library	1	Not installed
Google Play Billing Library, rev 5	5.0.0	Installed
Google Play Licensing Library	1	Not installed
Google Play Licensing Library, rev 2	2.0.0	Installed
Google Play services, rev 30	30.0.0	Installed
Google Repository	28	Installed

Construire un projet Firebase coté serveur

❑ Aller à <https://firebase.google.com/>. Cliquer sur le lien "ACCEDER A LA CONSOLE"

❑ On arrive à :



❑ Cliquer "CREATE NEW PROJECT"

❑ Dans la fenêtre de dialogue qui est affichée, indiquer un nom de projet et une région (pays)

❑ Remarque 1 : On peut mettre beaucoup d'applications Firebase dans un projet Firebase. Et on ne peut pas dépasser 4 projets Firebase même s'ils ont été détruits.

❑ Remarque 2 : il faut un compte gmail

Construire un projet Firebase pour Android

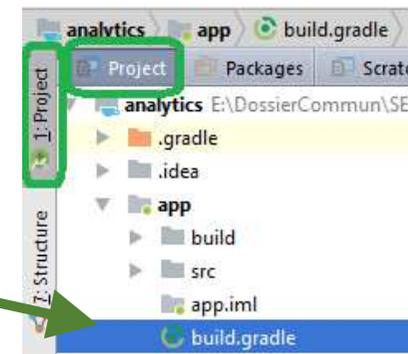
❑ Dans la fenêtre suivante indiquer que le projet sera utilisé par une application Android en cliquant sur le bouton



❑ Dans la fenêtre de dialogue qui apparaît, on peut entrer le nom du paquetage de l'Android app créée et son réduit SHA-1



❑ Pour trouver le paquetage de l'app analytics, dans Android Studio, se mettre dans la configuration `1:Project | Project` et ouvrir le `build.gradle` de l'app



❑ Le nom du paquetage est la valeur de `applicationId`

```
defaultConfig {  
    applicationId "com.google.firebase.quickstart.analytics"  
    .....
```

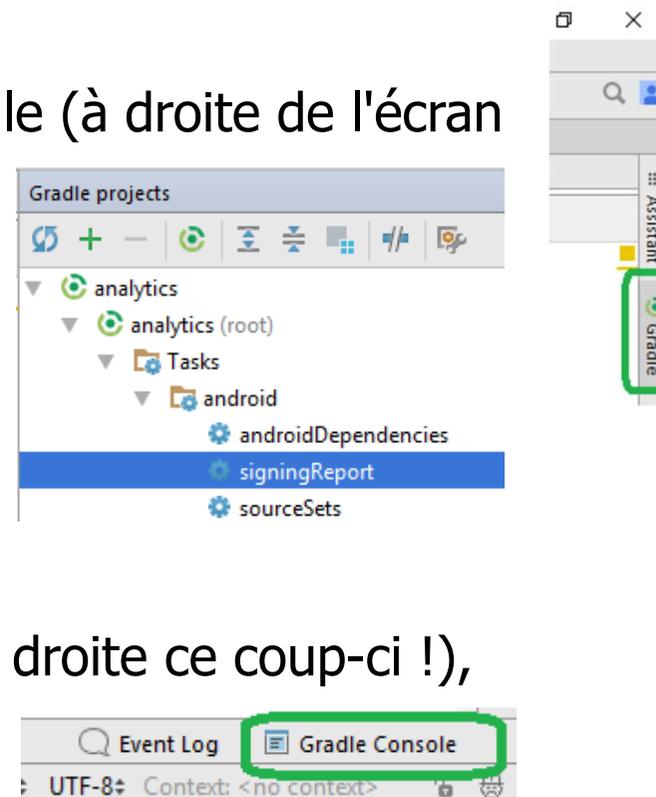
Le réduit SHA-1 avec Android Studio

- ❑ On peut obtenir ce réduit en ligne de commande : voir diapo suivante
- ❑ Avec Android Studio, cliquer Gradle (à droite de l'écran Android Studio) :

- ❑ Descendre dans signingReport
- ❑ Double cliquer sur signingReport

- ❑ Dans la Gradle Console (en bas à droite ce coup-ci !),

on a alors le réduit SHA-1



Le réduit SHA-1

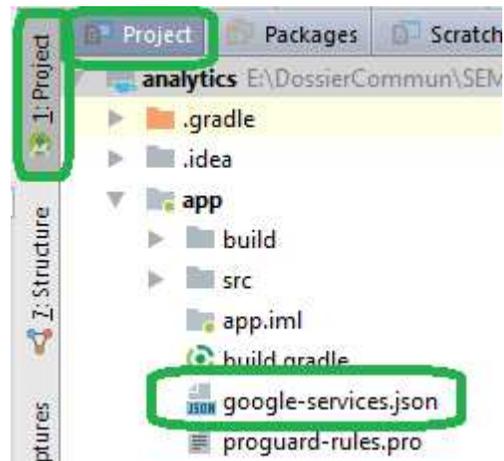
- ❑ Euh, il est optionnel, on peut donc s'en passer !
- ❑ Sinon, pour trouver le réduit SHA-1 de la clé de Debug écrire dans une console la commande :

```
keytool -exportcert -list -v -alias  
androiddebugkey -keystore  
repertoireContenantLEntrepotDeCles/debug.keystore  
> sovResult.txt
```
- ❑ *repertoireContenantLEntrepotDeCles* est souvent un chemin qui se termine par le répertoire `.android`
- ❑ Par exemple sous Unix
repertoireContenantLEntrepotDeCles = `~/ .android`
- ❑ On obtient donc (sans les gribouillis) :
- ❑ Reporter cette valeur dans la console Firebase (cf. diapo précédente)

```
Empreintes du certificat :  
MD5 : 7A:43:71:C5:7A:2B:AA  
SHA1 : DE:AF:73:0A:24:BD:5E:04:BD:57:B0:4A:0B:ED:0A:CB:73:C9:74:0C  
SHA256 : 82:0B:77:11:7A:8E:06:79:C3:97:BA:0B:E7:16:37:C6:DD:88:4E:E4:B4  
:14:DC:50:36:DB:07:50:03:80:89:CF  
Nom de l'algorithme de signature : SHA256withRSA  
Version : 3
```

L'Android app (1/2)

- Dans la console Firebase, il est indiqué de charger `google-services.json`. Le faire. Il faut l'installer sous app de l'app Android



L'Android app (2/2)

- ❑ En cliquant sur le bouton Continuer dans la console Firebase, on obtient :

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.

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

```
buildscript {
  dependencies {
    // Add this line
    classpath 'com.google.gms:google-services:3.2.0'
  }
}
```
2. Fichier `build.gradle` au niveau de l'application (`<project>/<app-module>/build.gradle`):

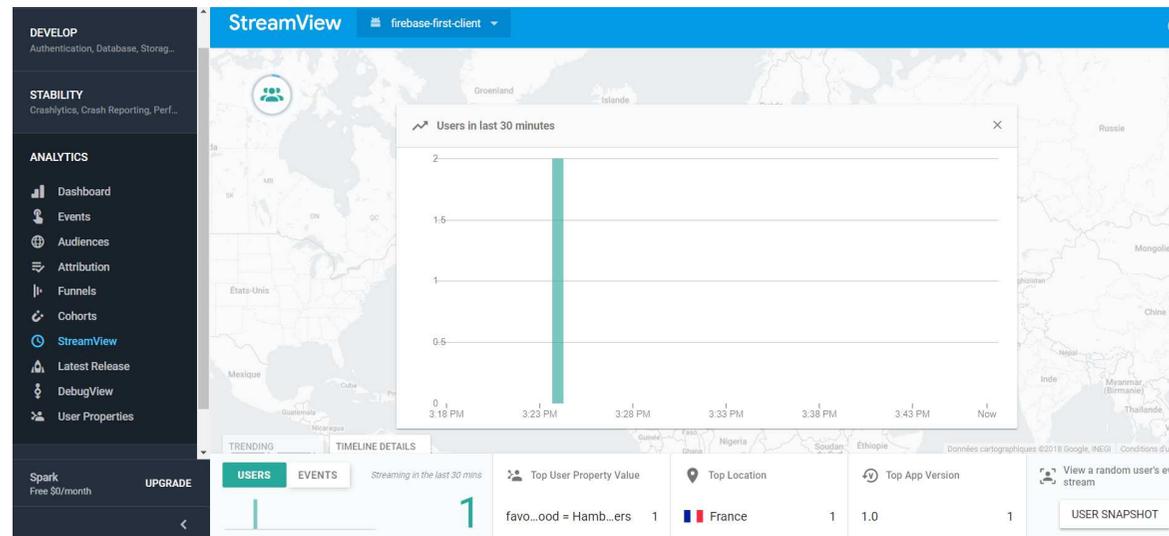
```
dependencies {
  // Add this line
  compile 'com.google.firebase:firebase-core:11.8.0'
}
...
// Add to the bottom of the file
apply plugin: 'com.google.gms.google-services'
```

Analytics inclus par défaut ⓘ

- ❑ Ce sont les ajouts qu'il faut faire (si ce n'est pas déjà fait) dans les (bons) `build.gradle`

Le projet Android-Firebase (1/3)

- ❑ En manipulant l'Android app, on doit obtenir côté Firebase (item StreamView :



- ❑ Hum Analytics est indicatif et les rafraichissements ne sont pas (= sont loin d'être) immédiats
- ❑ Mais avec un peu d'attente on peut avoir ...

Le projet Android-Firebase (2/3)

The screenshot displays the Firebase StreamView interface for the project 'firebase-first-client'. The interface includes a sidebar with navigation options under 'ANALYTICS' (Dashboard, Events, Audiences, Attribution, Funnels, Cohorts, StreamView, Latest Release, DebugView, User Properties) and 'DEVELOP' (Authentication, Database, Storage). The main area shows a map of Europe with a location pin over France. Below the map, there are filters for 'USERS' and 'EVENTS'. The 'EVENTS' filter is active, showing a count of 14. A dropdown menu is open, showing the event name 'user_engagement' and the filter 'fireba...creen = Id-A-A'. To the right, there is a table of app versions:

App version	Count	Percentage
1.0	14	100%

Below the table, there are filters for 'Top Location' (France) and 'Top App Version' (1.0). A 'USER SNAPSHOT' button is visible at the bottom right.

message de l'app Android

Le projet Android-Firebase (3/3)

- StreamView | Evénement | share_image | full_text affiche les messages envoyés

The screenshot displays the Firebase Analytics interface. On the left, a sidebar lists navigation options: STABILITY, ANALYTICS (Dashboard, Events, Audiences, Attribution, Funnels, Cohorts, StreamView, Latest Release, DebugView, User Properties), and Spark. The main area shows a map of the United States and a table of events. The table is titled 'Événements' and contains the following data:

Événement	Nombre	Pourcentage	Attribution	Classe	Classe	Classe
user_engagement	15	45,45 %	firebase_event_origin	I'd love you...ear about B	2	50 %
screen_view	7	21,21 %	firebase_screen_class	I'd love you...ear about C	1	25 %
select_content	6	18,18 %	firebase_screen	I'd love you...ear about D	1	25 %
share_image	4	12,12 %	firebase_screen_id			
session_start	1	3,03 %	full_text			
			image_name			

At the bottom, there is a filter bar with 'UTILISATEUR' and 'ÉVÉNEMENT' tabs, a search bar containing '60', and a filter for 'Principale zone géographique' set to 'France'.

Le code Android de Analytics (1/2)

- Tout part de :

```
<activity android:name="com.google.firebase.quickstart.analytics.MainActivity"
  ...
  <intent-filter>
    <action android:name="android.intent.action.MAIN"/>
    <category android:name="android.intent.category.LAUNCHER"/>
  </intent-filter>
</activity>
```

n'est ce pas !

- Et donc, une instance est créée sur laquelle est lancée :

```
private FirebaseAnalytics mFirebaseAnalytics;

protected void onCreate(Bundle savedInstanceState) {
  // Construction du début de l'interface graphique. Euh comment ?
  // Création d'un objet permettant de faire des log Firebase
  // et de positionner des propriétés

  mFirebaseAnalytics = FirebaseAnalytics.getInstance(this);

  // Complète l'IHM avec un PageViewer (et des fragments)
}
```

- Répondre aux questions ci-dessus

Le code Android de Analytics (2/2)

- Par la suite on envoie des messages Firebase Analytics par :

```
Bundle bundle = new Bundle();
bundle.putString(FirebaseAnalytics.Param.ITEM_ID, id);
bundle.putString(FirebaseAnalytics.Param.ITEM_NAME, name);
bundle.putString(FirebaseAnalytics.Param.CONTENT_TYPE, "image");
mFirebaseAnalytics.logEvent(FirebaseAnalytics.Event.SELECT_CONTENT, bundle);
```

- On a aussi un code qui est lancé qui est :

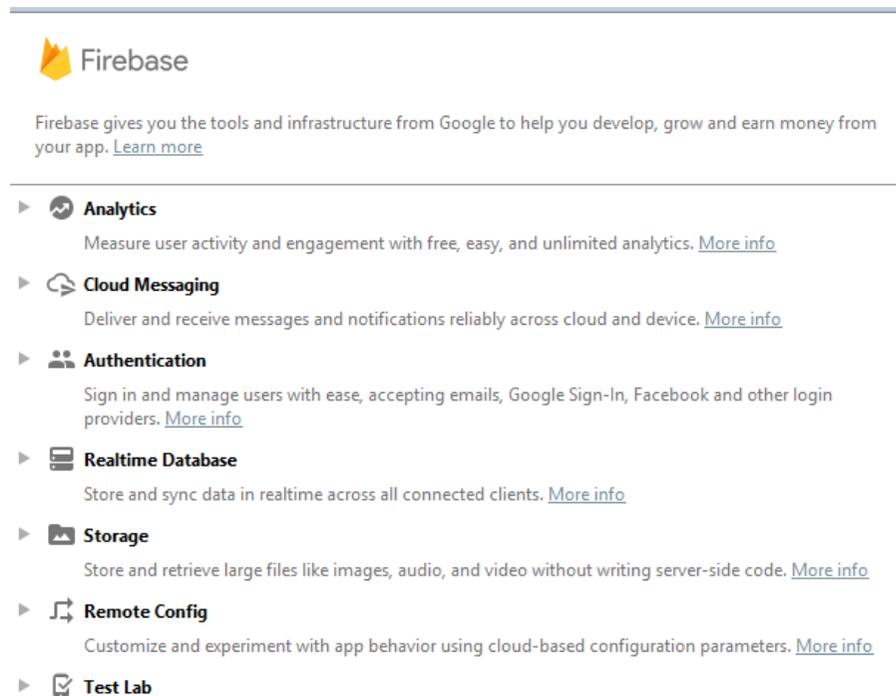
```
Bundle params = new Bundle();
params.putString("image_name", name);
params.putString("full_text", text);
mFirebaseAnalytics.logEvent("share_image", params);
```

Euh, quand et comment ?

- Pourquoi juste avant d'envoyer ce message au cloud Firebase Analytics, l'app Android demande à traiter en local ce message ? Choisir le presse-papier par exemple. Quelles lignes de code imposent de faire cela ?

Android - Firebase : une documentation

- ❑ Vérifier les indications sur les projets Firebase dans Android Studio par Tools | Firebase. On obtient :



- ❑ Et pour chaque projet, des indications (bonnes configs, code à utiliser, etc.)

Firestore Cloud Messaging : présentation

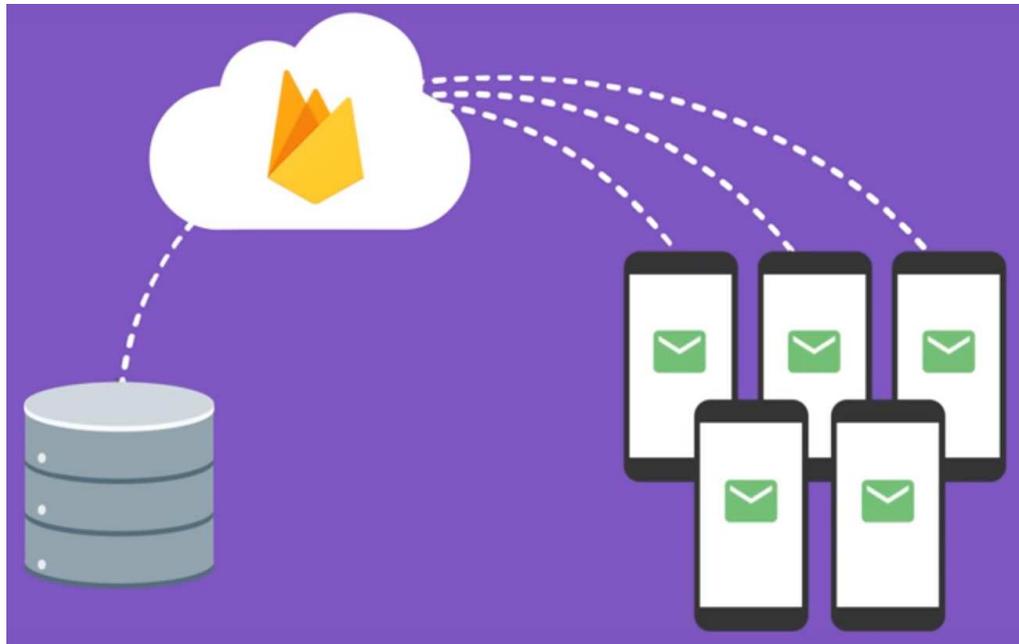
- ❑ Permet d'envoyer des messages du cloud vers des smartphones (Android, iOS) ou ordinateurs (par le web et JavaScript)



- ❑ Les messages peuvent être envoyés pour un smartphone, un groupe de smartphone, ou publiés sur un sujet
- ❑ Permet d'envoyer des messages de smartphones vers le cloud Firebase

Firebase Cloud Messaging : les étapes

- ❑ Enregistrer les smartphones auprès de Firebase dans le cloud
- ❑ Ecrire une app Firebase coté cloud
- ❑ Connecter l'ensemble :



Firebase Cloud Messaging : présentation

- ❑ Le but : "Deliver and receive messages and notifications reliably across cloud and device"
- ❑ Voir <https://firebase.google.com/docs/cloud-messaging/>
- ❑ Quand l'utilisateur clique sur le bouton "Log Token", l'environnement Firebase de l'app génère un "token" (qui est une `String`)
- ❑ Voir le code à <https://github.com/firebase/quickstart-android/tree/master/messaging>

Exercice



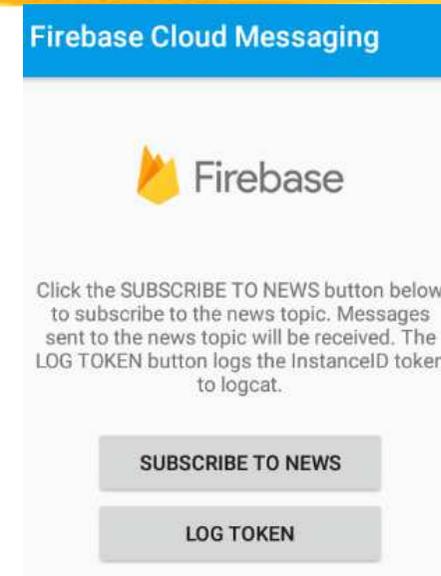
- Etudier Firebase Cloud Messaging

Projet Android - Firebase Cloud Messaging (1/4)

❑ Pour récupérer ce token, il suffit de créer un service qui est une sous-classe de `FirebaseInstanceIdService`. Lorsque le token est construit, la méthode `public void onTokenRefresh()` de cette classe est lancée.

❑ On récupère alors le token par

```
String leToken = FirebaseInstanceId.getInstance().getToken();
```



Projet Android - Firebase Cloud Messaging (2/4)

- ❑ Ce token identifie le smartphone
- ❑ On peut alors l'envoyer au projet Firebase
- ❑ Bref on a :

```
public class MaClasseRecupToken extends FirebaseInstanceIdService {  
    public void onTokenRefresh(){  
        String leToken = FirebaseInstanceId.getInstance().getToken();  
        envoiTokenPourEtreIdentifie(leToken);  
    }  
}
```

Projet Android - Firebase Cloud Messaging (3/4)

- ❑ Pour envoyer ou recevoir des messages dans le cloud, il suffit de créer un service qui hérite de `FirebaseMessagingService`
- ❑ Euh service = ?
- ❑ On doit forcément (!) enregistrer le service dans le `AndroidManifest.xml` :

```
<service
  android:name=".MaClasseRecupToken">
  <intent-filter>
    <action android:name="com.google.firebase.INSTANCE_ID_EVENT"/>
  </intent-filter>
</service>
```

Projet Android - Firebase Cloud Messaging (4/4)

- ❑ Par la suite lorsque le cloud Firebase envoie des messages au smartphone, celui-ci doit en être averti. Pour cela, évidemment un service sensible à un Intent !

- ❑ Enregistré comme :

```
<service
  android:name=".MyFirebaseMessagingService">
  <intent-filter>
    <action android:name="com.google.firebase.MESSAGING_EVENT"/>
  </intent-filter>
</service>
```

- ❑ MyFirebaseMessagingService est une classe qui hérite de `com.google.firebase.messaging.FirebaseMessagingService`
- ❑ Lorsqu'un message est reçu, la méthode `public void onMessageReceived(RemoteMessage remoteMessage)` de ce service est lancée (cours sur les services)
- ❑ Par la suite, ce service peut lancer des "jobs" divers et variés suivant le message reçu



Fin