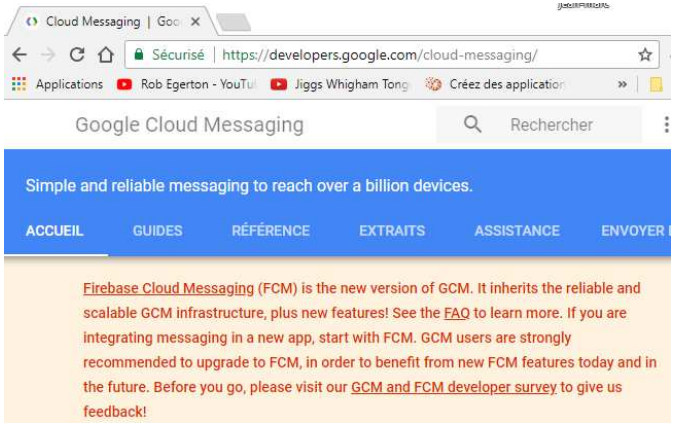
Android et le cloud

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 (Envolve) 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 débuter

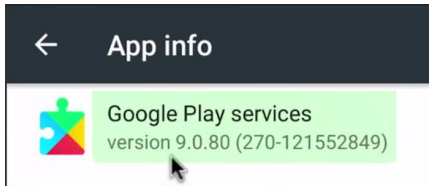
- La vidéo qui se trouve à `https://firebase.google.com/docs/android/setup?authuser=0` (Add Firebase to Your Android Project) ou bien `https://www.youtube.com/watch?v=cNPCgJW8c-E` fonctionne
- 3 étapes :
  - vérifier les numéros de version de parties d'Android Studio
  - créer un projet Firebase
  -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

Firebase et Android : des exemples

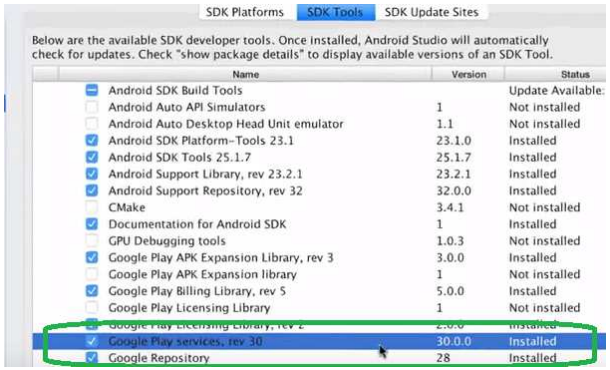
- Récupérer les exemples de `https://github.com/firebase/quickstart-android`
- Dézipper
- Dans Android Studio charger (Close Project puis "Open an existing Android Studio project"), l'app Analytics

Numéros de version de parties d'Android Studio

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



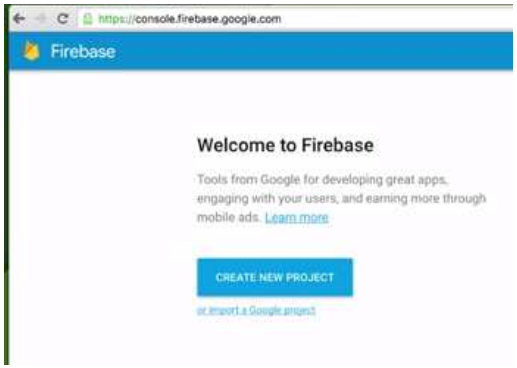
- Vérifier qu'Android Studio a installé :
  - Google Play Services de version >= 30.
    Pour cela SDK Manager | onglet SDK Tools | Google Play Services
  - Google repository de version >= 26



- On est prêt à écrire une Android app utilisant Firebase

Construire un projet Firebase coté serveur

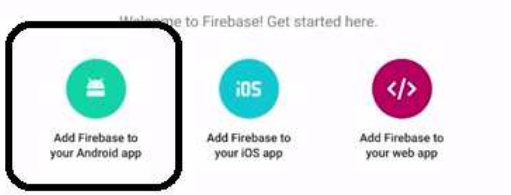
- Aller à `https://firebase.google.com/`. Cliquer sur le lien "ACCEDER A LA CONSOLE"
- On arrive à :



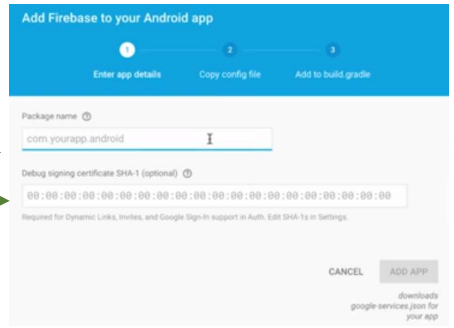
- Cliquer "CREATE NEW PROJECT"
- Dans la fenêtre de dialogue qui est affichée, indiquer un nom de projet et une région (pays)
- Remarque 1 : On peut mettre beaucoup d'applications Firebase dans un projet Firebase. Et on ne peut pas dépasser 4 projets Firebase même s'ils ont été détruits.
- Remarque 2 : il faut un compte gmail

Construire un projet Firebase pour Android

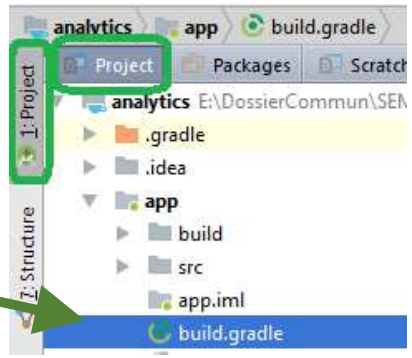
- Dans la fenêtre suivante indiquer que le projet sera utilisé par une application Android en cliquant sur le bouton



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



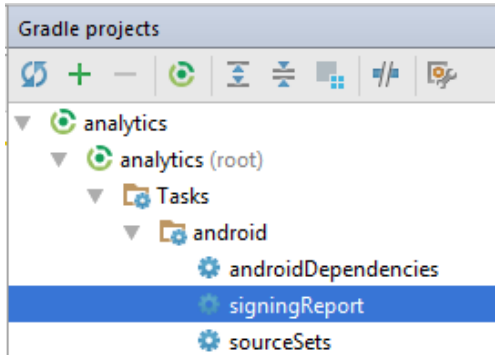
- Pour trouver le paquetage de l'app `analytics`, dans Android Studio, se mettre dans la configuration `1:Project | Project` et ouvrir le `build.gradle` de l'app
- Le nom du paquetage est la valeur de `applicationId`



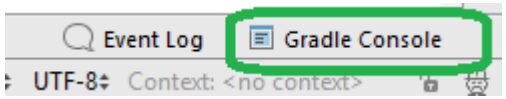

```
defaultConfig {
    applicationId "com.google.firebase.quickstart.analytics"
```

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) :



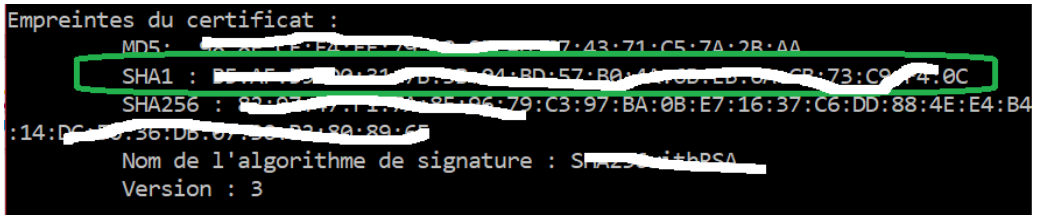
- Descendre dans signingReport
- Double cliquer sur signingReport
- Dans la Gradle Console (en bas à droite ce coup-ci !),



on a alors le réduit SHA-1

Le réduit SHA-1

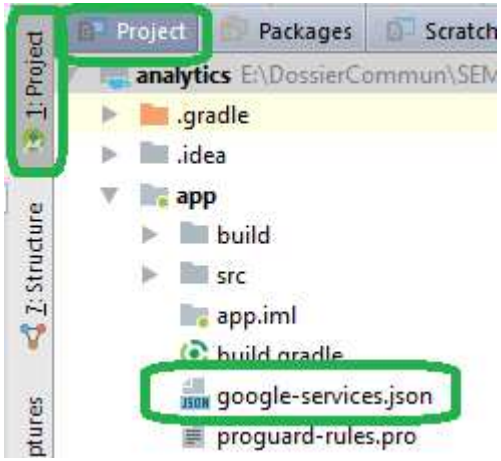
- Euh, il est optionnel, on peut donc s'en passer !
- Sinon, pour trouver le réduit SHA-1 de la clé de Debug écriredans une une console la commande :
  `keytool -exportcert -list -v -alias androiddebugkey -keystore` *`repertoireContenantLEntrepotDeCles`*`/debug.keystore > sovResult.txt`
- *`repertoireContenantLEntrepotDeCles`* 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)

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`) :

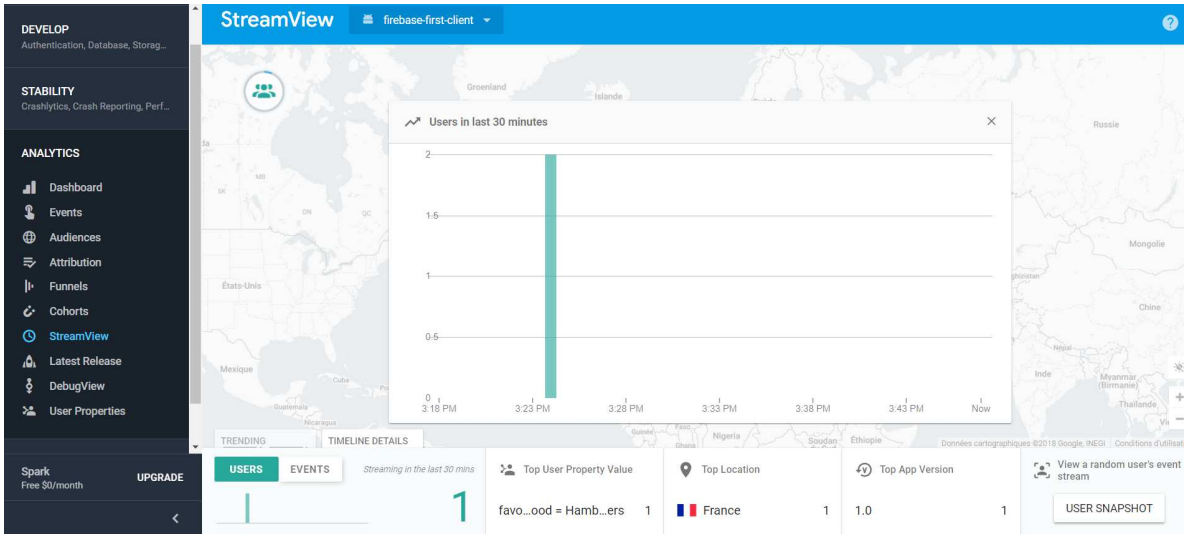
```
dependencies {
  // Add this line
  compile 'com.google.firebase:firebase-core:11.8.0'
}
...
// Add to the bottom of the file
apply plugin: 'com.google.gms.google-services'
```

Analytics inclus par défaut

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

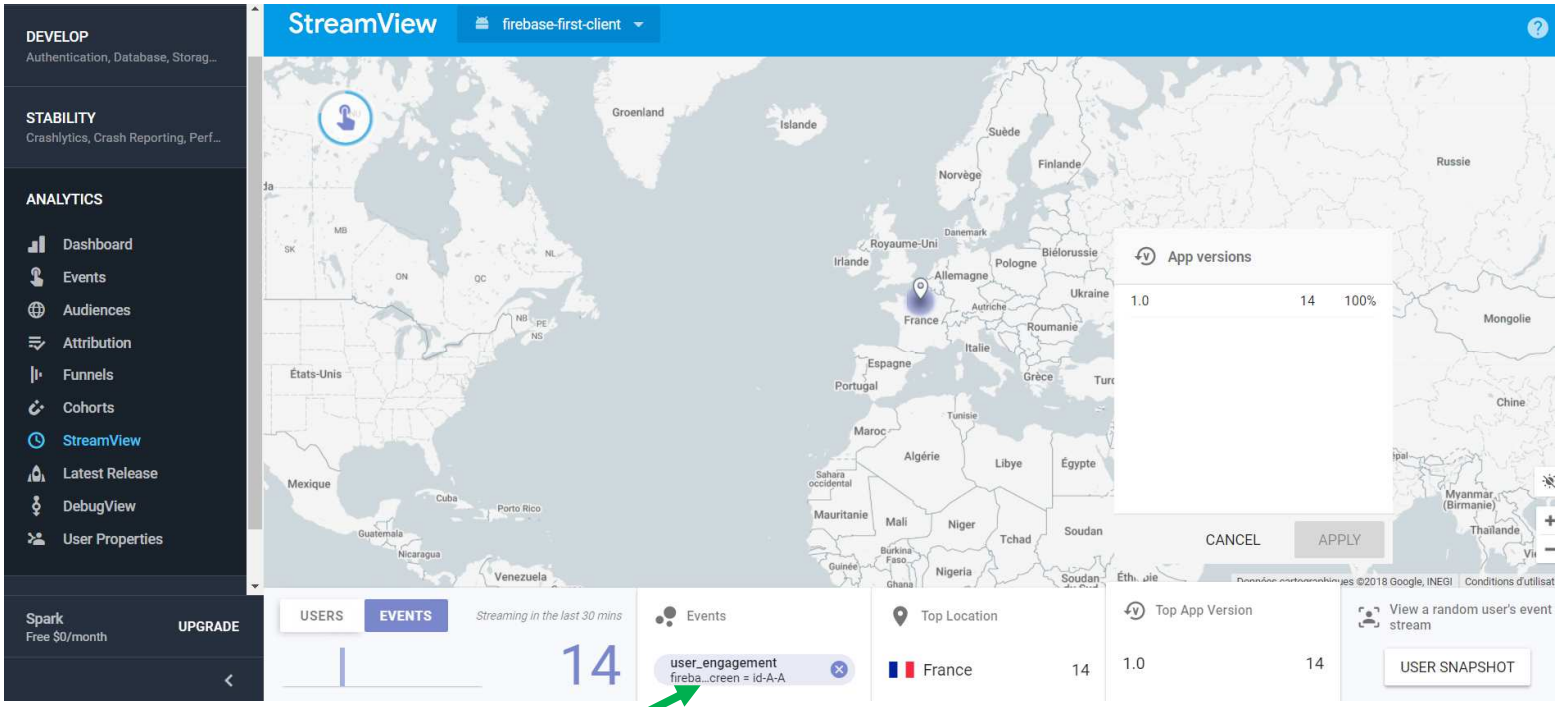
Le projet Android-Firebase (1/3)

- En manipulant l'Android app, on doit obtenir coté 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 ...

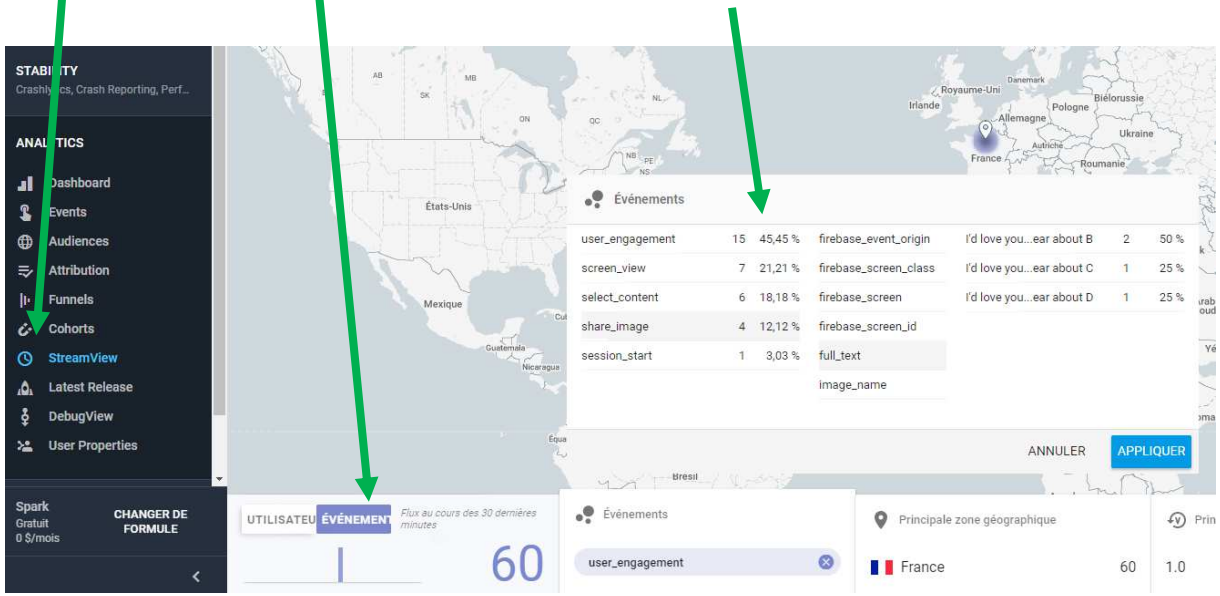
Le projet Android-Firebase (2/3)



- message de l'app Android

Le projet Android-Firebase (3/3)

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



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 :

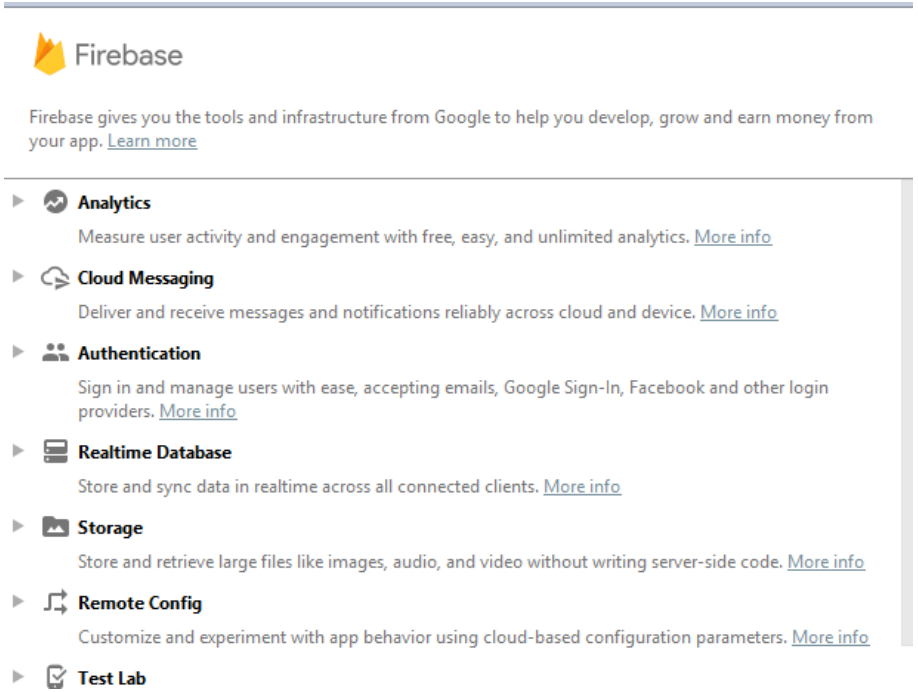
```
Bundle params = new Bundle();
params.putString("image_name", name);
params.putString("full_text", text);
mFirebaseAnalytics.logEvent("share_image", params);
```

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.)

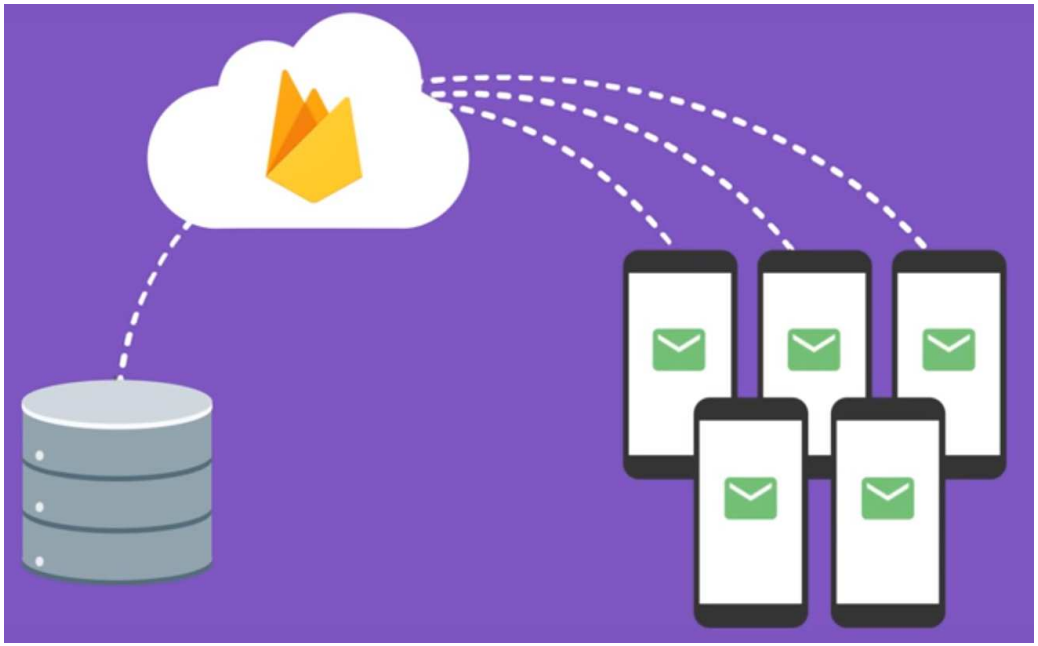
Firebase Cloud Messaging : présentation

- 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

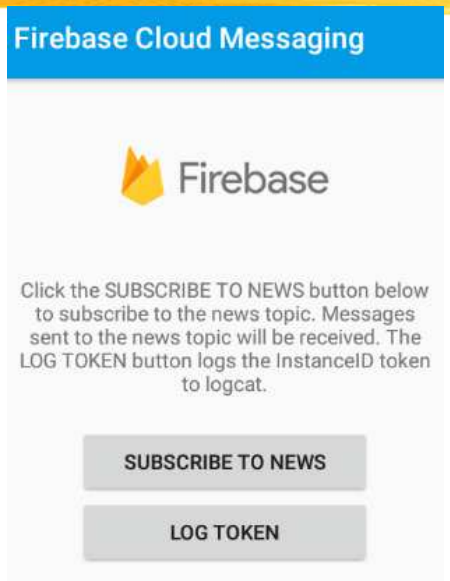
- Le but : "Deliver and receive messages and notifications reliably across cloud and device"
- Voir `https://firebase.google.com/docs/cloud-messaging/`
- Quand l'utilisateur clique sur le bouton "Log Token", l'environnement Firebase de l'app gènère un "token" (qui est une `String`)
- Voir le code à `https://github.com/firebase/quickstart-android/tree/master/mess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

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 (!) enregistré 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 celà, 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