1. 웹뷰 보완

웹에서 사용자 계정 등록시 해당 사용자의 어플로 푸쉬 전달해야함.

혹시나 해서 Service를 하나 만들어서 무한 반복문 (백그라운드로 서버로 부터 메시지가 오는지 체크하기 위해서)를 돌려 봤는데 앱이 터짐

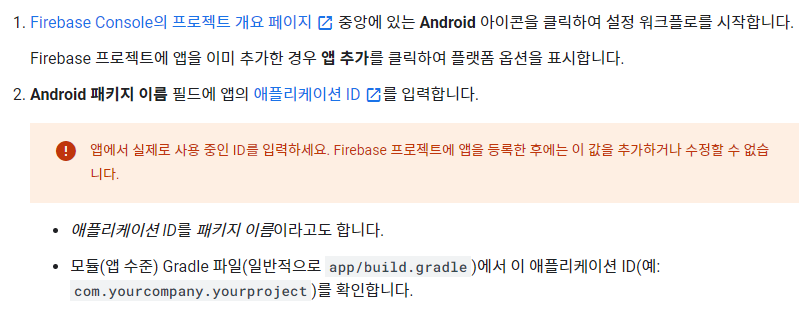
그래서 찾아본 결과 게임이나 여타 게임들의 경우 푸쉬를

파이어 베이스 기반의 Google Cloud Message를 사용함을 알게 되었음.

================================================================

**firebase 프로젝트 만들기**

1. Firebase 프로젝트 생성
2. Firebase에 앱 등록



-----------------------여기까지가 선배님이 해주시면 되는거------아래는 내가해야되는거----------

1. Firebase ANdroid 구성 파일을 추가.
   1. **Download google-services.json(google-services.json 다운로드)**을 클릭하여 Firebase Android 구성 파일(google-services.json)을 가져옵니다.
   2. google-service 플러그인을 Gradle에 추가함.
      1. build.gradle에 아래와 같이 저장.

buildscript {

repositories {

// Check that you have the following line (if not, add it):

google() // Google's Maven repository

}

dependencies {

// … // Add the following line:

classpath 'com.google.gms:google-services:4.2.0' // Google Services plugin

}

}

allprojects {

// ...

repositories {

// Check that you have the following line (if not, add it):

google() // Google's Maven repository

// ...

}

}

* + 1. app/build.gradle의 파일 하단에 다음 내용을 추가

apply plugin: 'com.android.application'

android {

// ...

}

// Add the following line to the bottom of the file:

apply plugin: 'com.google.gms.google-services' // Google Play services Gradle plugin

4. Firebase SDK(Firebase 제품)등을 Android앱에 추가

app / build.gradle에

dependencies {

// ...

implementation 'com.google.firebase:firebase-core:17.0.0'

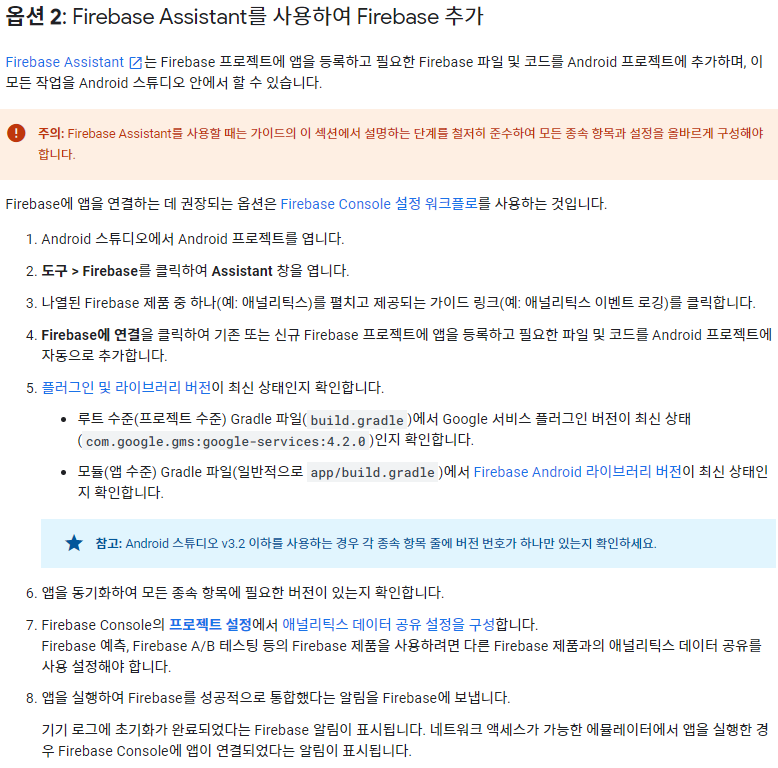
// Getting a "Could not find" error? Make sure that you've added

// Google's Maven repository to your root-level build.gradle file

}

를 추가함.

------------------------------------------ 이러한 복잡하고 긴 과정 대신 아래 같은 방법으로도 가능하다.



여기까지 완료했다면

android에서 client 설정을 하고 구현해서 쓰면 된다.

**client 설정하기**

앱의 매니페스트에 다음을 추가합니다

FirebaseMessagingService를 확장하는 서비스를 추가합니다.

-> 포그라운드 앱의 알림 수신, 데이터 페이로드 수신, 업스트림 메시지 전송 등을 수행하려면 이 서비스를 확장해야 합니다.

<service

android:name=".java.MyFirebaseMessagingService"

android:exported="false">

<intent-filter>

<action android:name="com.google.firebase.MESSAGING\_EVENT" />

</intent-filter>

</service>

(AndroidManifest.xml의 전문은 여기에 있습니다.<https://github.com/firebase/quickstart-android/blob/8a3169ae7f75e38665a62c520ccf8960609ab815/messaging/app/src/main/AndroidManifest.xml#L50-L56>)

푸시 알림 아이콘과 색상을 변경하려면 아래 부분을 변경하면 되고,

<!-- Set custom default icon. This is used when no icon is set for incoming notification messages.

See README(https://goo.gl/l4GJaQ) for more. -->

<meta-data

android:name="com.google.firebase.messaging.default\_notification\_icon"

android:resource="@drawable/ic\_stat\_ic\_notification" />

<!-- Set color used with incoming notification messages. This is used when no color is set for the incoming

notification message. See README(https://goo.gl/6BKBk7) for more. -->

<meta-data

android:name="com.google.firebase.messaging.default\_notification\_color"

android:resource="@color/colorAccent" />

알림 채널을 직접 만들어서 사용하려면

<meta-data android:name="com.google.firebase.messaging.default\_notification\_channel\_id"

android:value="@string/default\_notification\_channel\_id" />

이 부분을 수정해서 사용하면 된다.

이것들을 수정 하지 않아도 작동에는 문제 x

**토큰 엑세스**

단일 기기를 타겟팅하거나 기기 그룹을 만들려면 [FirebaseMessagingService](https://firebase.google.com/docs/reference/android/com/google/firebase/messaging/FirebaseMessagingService?hl=ko)를 확장하고 onNewToken을 재정의하여 이 토큰에 액세스해야 합니다.

현재 토큰 검색

FirebaseInstanceId.getInstance().getInstanceId()

.addOnCompleteListener(new OnCompleteListener<InstanceIdResult>() {

@Override

public void onComplete(@NonNull Task<InstanceIdResult> task) {

if (!task.isSuccessful()) {

Log.w(TAG, "getInstanceId failed", task.getException());

return;

}

// Get new Instance ID token

String token = task.getResult().getToken();

// Log and toast

String msg = getString(R.string.msg\_token\_fmt, token);

Log.d(TAG, msg);

Toast.makeText(MainActivity.this, msg, Toast.LENGTH\_SHORT).show();

}

});

토큰 생성 모니터링

/\*\*

\* Called if InstanceID token is updated. This may occur if the security of

\* the previous token had been compromised. Note that this is called when the InstanceID token

\* is initially generated so this is where you would retrieve the token.

\*/

@Override

public void onNewToken(String token) {

Log.d(TAG, "Refreshed token: " + token);

// If you want to send messages to this application instance or

// manage this apps subscriptions on the server side, send the

// Instance ID token to your app server.

sendRegistrationToServer(token);

}

토큰이 확보되었으면 앱 서버로 전송하고 원하는 방법으로 저장할 수 있습니다.

## 

## **Google Play 서비스 확인**

Play 서비스 SDK를 사용하는 앱은 Google Play 서비스 기능에 액세스하기 전에 항상 기기에 호환되는 Google Play 서비스 APK가 있는지 확인해야 합니다.

기기에 호환되는 Google Play 서비스 버전이 없으면 앱에서 [GoogleApiAvailability.makeGooglePlayServicesAvailable()](https://developers.google.com/android/reference/com/google/android/gms/common/GoogleApiAvailability.html?hl=ko#public-methods)을 호출하여 사용자가 Play 스토어에서 Google Play 서비스를 다운로드하도록 허용할 수 있습니다

## **자동 초기화 방지**

인스턴스 ID가 자동 생성되는 것을 방지하려면 AndroidManifest.xml에 다음 메타데이터 값을 추가하여 FCM 및 애널리틱스 두 곳에서 모두 자동 초기화를 사용 중지해야 합니다.

<meta-data

android:name="firebase\_messaging\_auto\_init\_enabled"

android:value="false" />

<meta-data

android:name="firebase\_analytics\_collection\_enabled"

android:value="false" />

FCM을 재사용 하려면

FirebaseMessaging.getInstance().setAutoInitEnabled(true);