Recibir notificaciones

1. Tener un proyecto en Firebase con la App añadida.

2. Tener la librería FCM

```
implementation 'com.google.firebase:firebase-messaging'
```

3. Crear una clase que herede de FirebaseMessagingService

```
public class MyFirebaseMessagingService extends FirebaseMessagingService {
```

4. Añadir la clase anterior en el Manifest, como un servicio. Con el filtro MESSAGING_EVENT podrá recibir notificaciones de Firebase.

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

5. En la clase del paso 3, sobreescribir el método onMessageReceived.
Este método es el que recibe los mensajes de Firebase. Si el mensaje recibido es de datos, podemos coger la información que se recibido y mostrar una notificación con dicha información.

```
@Override
public void onMessageReceived(RemoteMessage remoteMessage) {


    // comprueba si el mensaje es de datos
    if (remoteMessage.getData().size() > 0) {

        Log.d(TAG,  msg: "Datos del mensaje: " + remoteMessage.getData());
        Map<String, String> datos = remoteMessage.getData();
        String titulo = datos.get("title");
        String body = datos.get("body");
        String fecha = datos.get("fecha");
        sendNotification(titulo, body, fecha);
    }
```

6. El método sendNotification mostraría una notificación(Tema 6.2. Notificaciones) con el título, body y fecha recibidos de Firebase.

Generar y enviar token
Hay que generar el token del dispositivo y subirlo en la base de datos, a una colección de tokens. De esta forma, esta colección es accesible por la App a la hora de enviar un mensaje y, por tanto, una notificación.

1. Generar token y almacenarlo de alguna forma para tenerlo disponible en cualquier momento. Como el método tarda un poco en ejecutarse, recomiendo pedirlo al hacer login.

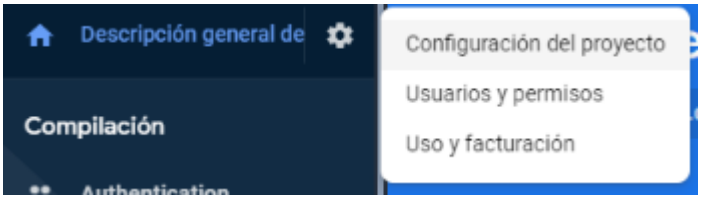
```
FirebaseMessaging.getInstance().getToken()
        .addOnCompleteListener(new OnCompleteListener<String>() {
            @Override
            public void onComplete(@NonNull Task<String> task) {
                if (!task.isSuccessful()) {
                    Log.w(TAG,  msg: "Fetching FCM registration token failed", task.getException());
                    return;
                }

                // Get new FCM registration token
                String token = task.getResult();
                editor.putString(Constants.DB_USER_TOKEN, token);
                editor.apply();
            }
        });
```

2. Cuando creamos necesario, añadir este token a la base de datos. Dependerá de cómo hayamos estructurado el tema tokens. Por ejemplo, si vamos a tener una colección de token para cada foro, pues al añadir un mensaje de forma exitosa, añadimos el token a dicha colección.

```
String token = preferences.getString(Constants.DB_USER_TOKEN,  defValue: "");
if (!token.isEmpty()) {
    HashMap<String, String> hashMap = new HashMap<>();
    hashMap.put(Constants.DB_USER_TOKEN, token);
    esteForo.collection(Constants.DB_USER_TOKEN).document(token).set(hashMap);
}
```

Enviar notificaciones
Como ya se comentó, vamos a enviar notificaciones desde la App saltándonos el paso de que exista una aplicación de servidor que lo haga.

1. Entrar en la consola de Firebase, en la configuración de nuestro proyecto



2. En la pestaña Cloud Messaging, copiar la Clave del Servidor

3. Pegarla en la clase con la petición Retrofit para mandar notificaciones

```
public interface MyClient {

    String BASE_URL = "https://fcm.googleapis.com/";
    String SERVER_TOKEN = "AAAAvuahWA0:APA91bHlBtw9RZMAOOwLt1SGmheXdECl3ekVjy9gonYqQy

    @Headers({"Authorization:key=" + SERVER_TOKEN, "Content-Type:application/json"})
    @POST("fcm/send")
    Call<Object> sendNotification(@Body MyNotification mynotif);
}
```

4. Con POJO Schema, sacar las clases necesarias para enviar una notificación. El formato en el que se mandan las notificaciones es éste, donde registration_ids debe ser un ArrayList con los token, y data es un objeto que puede tener todos los datos que queramos enviar.
La clase MyNotification del @Body del punto anterior se corresponde con la información de este JSON.

```
{
"registration_ids" :
   ["f_sGsl4MQAOOPk7WRISOxb:APA91bHnhvkKbYnco7z2L5jEw53Q-QR
   oiroAeml H62wZ6PBDdf557mn644GDXnnrI_D4ozft","asfdasdf"],
"data" : {
    "body" : "Notification Body",
    "title": "Notification Title",
    "key_1" : "Value for key_1",
    "key_2" : "Value for key_2"
}
}
```

5. Al añadir un mensaje a la base de datos con éxito, obtenemos el listado con todos los token de los usuarios a los que se va a mandar la notificación

```
private void sendNotification(String title, String msg) {
    CollectionReference usuarios = esteForo.collection(Constants.DB_USER_TOKEN);
    usuarios.get().addOnSuccessListener(new OnSuccessListener<QuerySnapshot>() {
        @Override
        public void onSuccess(QuerySnapshot queryDocumentSnapshots) {
            List<String> tokens = new ArrayList<>();

            for (DocumentSnapshot document : queryDocumentSnapshots) {
                Map<String, Object> map = document.getData();
                tokens.add((String) map.get("token"));
            }
                sendNotifReal(title, msg, tokens);
        }
    });
}
```

6. El método sendNotifReal manda la notificación usando Retrofit a la lista de token obtenida

```
private void sendNotifReal(String title, String msg, List<String> token) {

    SimpleDateFormat simpleDateFormat = new SimpleDateFormat( pattern: "dd-MM-yyyy-hh-mm-ss");
    String fecha = simpleDateFormat.format(new Date());

    NotifData notifData = new NotifData(msg, title, fecha);
    MyNotification notif = new MyNotification(token, notifData);

    Retrofit retrofit = new Retrofit.Builder()
            .baseUrl(MyClient.BASE_URL)
            .addConverterFactory(GsonConverterFactory.create())
            .build();

    MyClient myClient = retrofit.create(MyClient.class);
    Call<Object> objectCall = myClient.sendNotification(notif);
    objectCall.enqueue(new Callback<Object>() {
        @Override
        public void onResponse(Call<Object> call, Response<Object> response) {
            if (response.isSuccessful()) {
                Object object = response.body();
                try {
```