Firebase Cloud Messaging (FCM) Migration Guide

1. Introduction to Firebase Cloud Messaging (FCM)

Firebase Cloud Messaging (FCM) is a cross-platform messaging solution that allows you to reliably send messages at no cost. FCM enables you to notify client apps that new data is available to sync or send notification messages to drive user engagement and retention. For use cases such as instant messaging, a message can transfer a payload of up to 4096 bytes to a client app.

---

2. Migrating from Legacy FCM APIs to FCM HTTP v1

Apps using the deprecated FCM legacy APIs for HTTP and XMPP should migrate to the HTTP v1 API as soon as possible. These legacy APIs were deprecated on **June 20, 2023**, and shutdown begins on **July 22, 2024**.

2.1 Update the Server Endpoint

The endpoint URL for the HTTP v1 API differs from the legacy endpoint in the following ways:

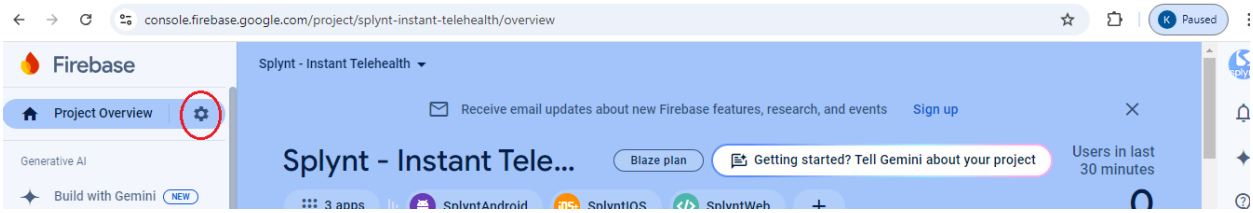
- The path includes the project ID of your Firebase project in the format /projects/myproject-ID/. This project ID can be found in the "General Project Settings" tab of the Firebase console.
- It explicitly specifies the send method as :send.
- **Old HTTP Request:**
  POST https://fcm.googleapis.com/fcm/send (deprecated)
- **New HTTP v1 Request URL:**
  POST https://fcm.googleapis.com/v1/projects/{projectId}/messages:send

2.2 How to Get Your Project ID:

1. Log in to your Firebase account.
2. Click on **“Go to Console”** and select your project.

Created By
Karmanye Upadhyay

3. Click on the settings icon and navigate to **Project Settings**.



4. Your **Project ID** will be displayed here—copy it and replace {projectId} in the new URL.

---

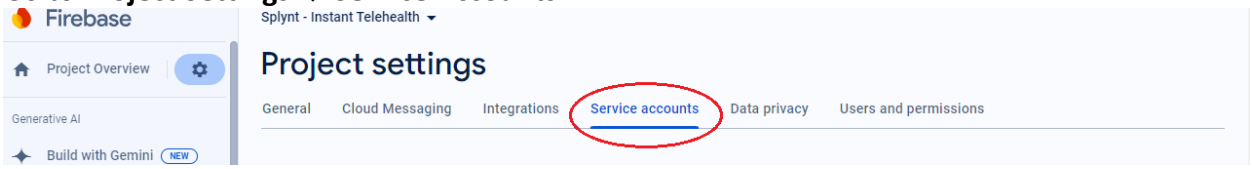
3. Updating Authorization of Send Requests

In place of the server key string used in legacy requests, HTTP v1 send requests require an **OAuth 2.0 access token**.

3.1 Generating the Auth Token

To generate the OAuth 2.0 access token, you will need to download a JSON file for your service account. Follow these steps:

1. Go to **Project Settings** → **Service Accounts**.



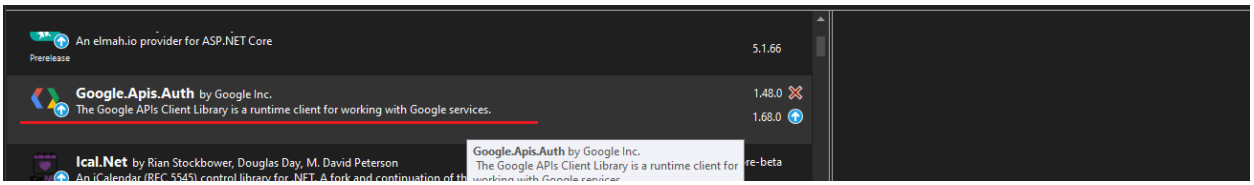
2. Click **Generate New Private Key**, then confirm by clicking **Generate Key**.
3. Securely store the downloaded JSON file containing the key.

---

4. Integration in Code

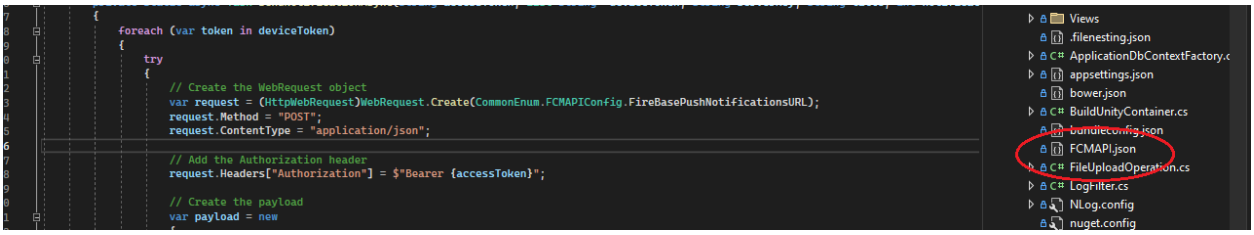
4.1 Installing Required NuGet Package

To integrate with FCM, install the NuGet package called **"Google.Apis.Auth"** in your project.



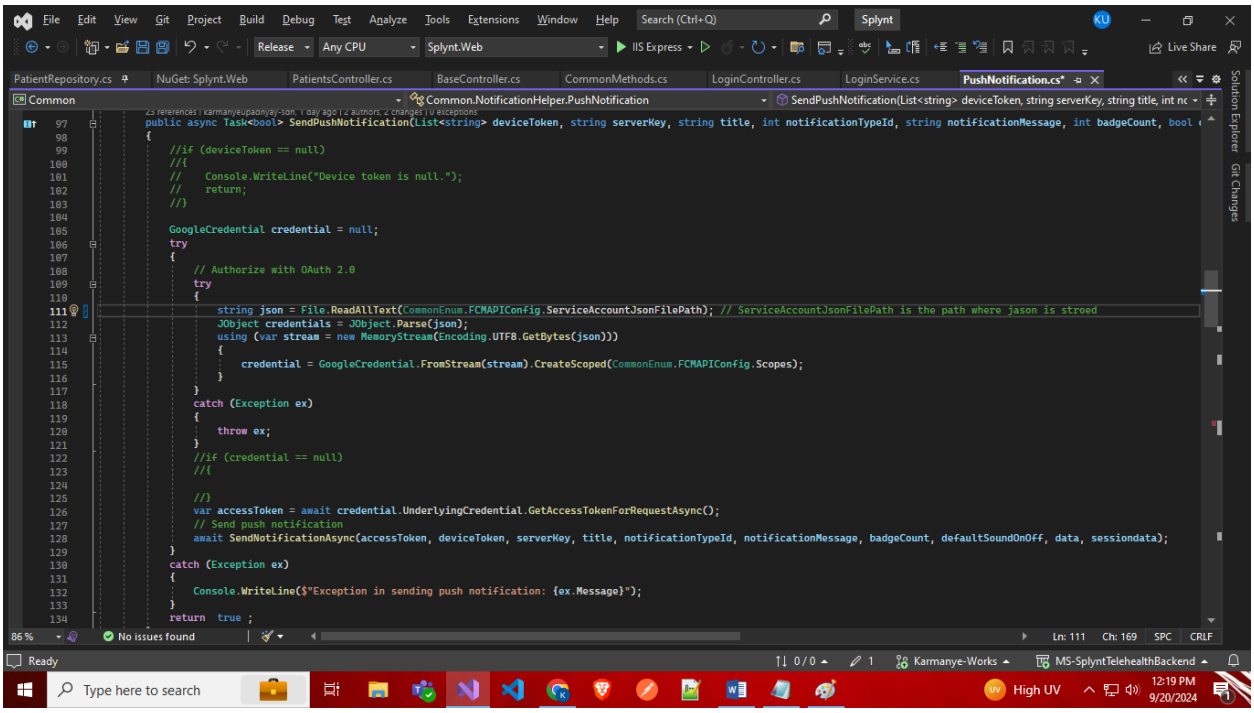
4.2 Creating a JSON File for Service Account

1. Save the JSON text (downloaded earlier) into a new .json file in your project.
2. This file will be used to authenticate with FCM.



4.3 Creating the Access Token Service

This service is responsible for generating the OAuth 2.0 access token from the service account JSON file:



Reference code

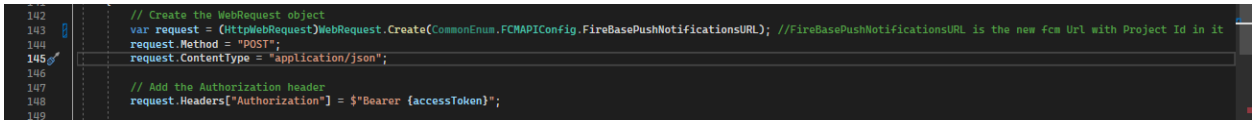
- **What was done:**
    - The service reads the FCM service account credentials from the JSON file.
    - The credentials are parsed and used to create a GoogleCredential object with the required OAuth 2.0 scopes.
    - The access token is retrieved using the GetAccessTokenForRequestAsync method.

Once the access token is obtained, the second service for generating and sending the notification payload is called.

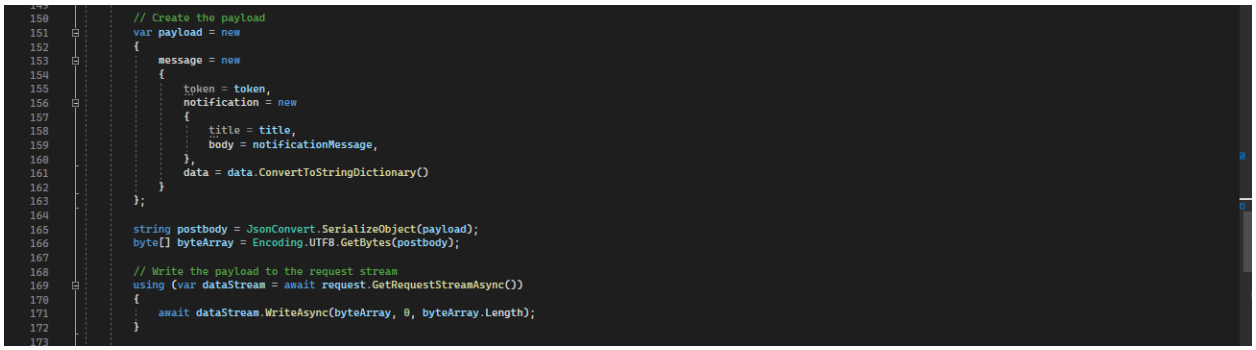
4.4 Sending Push Notifications

The second service handles generating and sending the notification payload to FCM:

1. **Create the WebRequest Object**:
    - This object contains the new FCM URL, method, content type, and the auth token obtained from the access token service.



2. **Create the Payload**:
    - Generate the payload. Ensure that all values (such as integers, booleans, and lists) are converted to **strings**, as FCM only accepts string values in the payload.



3. **Send the Request and Handle the Response**:
    - Make a POST request to FCM with the payload serialized into a JSON string.
    - Log the server response for future reference or debugging.

```
        // Get the response from the server
        using (var response = (HttpWebResponse)await request.GetResponseAsync())
        {
            using (var reader = new StreamReader(response.GetResponseStream()))
            {
                string responseBody = await reader.ReadToEndAsync();
                Console.WriteLine($"Response from server for token {deviceToken}: " + responseBody);
            }
        }
    }
    catch (WebException ex)
    {
        Console.WriteLine($"Failed to send notification to token {deviceToken}. Status code: {((HttpWebResponse)ex.Response)?.StatusCode}");
        using (var reader = new StreamReader(ex.Response.GetResponseStream()))
        {
            string errorResponse = await reader.ReadToEndAsync();
            Console.WriteLine("Error response: " + errorResponse);
        }
    }
```

4. **Error Handling**:
    - If any error occurs, the code logs the error response and HTTP status code for further troubleshooting.

Summary of Key Changes from Legacy API

- **OAuth 2.0 Authorization**: The legacy server key has been replaced by an OAuth 2.0 access token.
- **Endpoint URL Update**: The new endpoint URL now includes the project ID and the /messages:send path.
- **Payload Structure**: Ensure all payload values are converted to strings, as FCM's new API only accepts string-based properties.

This document should guide you through migrating from the legacy FCM APIs to the newer HTTP v1 API and integrating it into your code.
