



*Marco Cantu* 30 May 2019

Last year, Google deprecated Google Cloud Messaging, also known as GCM. We have built-in support in FireMonkey for GCM.

Google has [announced](#) that the "The GCM server and client APIs are deprecated and will be removed as soon as May 29, 2019. Migrate GCM apps to [Firebase Cloud Messaging](#) (FCM), which inherits the reliable and scalable GCM infrastructure, plus many new features."

In order to use push notification support in your FireMonkey Android applications going forward, you will need to use Google's Firebase. In this blog post, we're going to cover the steps required to add Firebase push notification support to your FireMonkey Android applications with Delphi, C++Builder and RAD Studio 10.3.1.

Before you get started, download the Firebase FMX package from the GetIt Package Manager in the IDE (Tools > GetIt Package Manager). You can find it in the "Tools" category.



As you can see in our [latest product roadmap](#), we intend to simplify this support further in the upcoming 10.3.2 release (also adding support for Android Firebase push notifications in RAD Server) and provide fully integrated support for Firebase and additional related services in 10.4. Please note that features are not committed until completed and GA released.

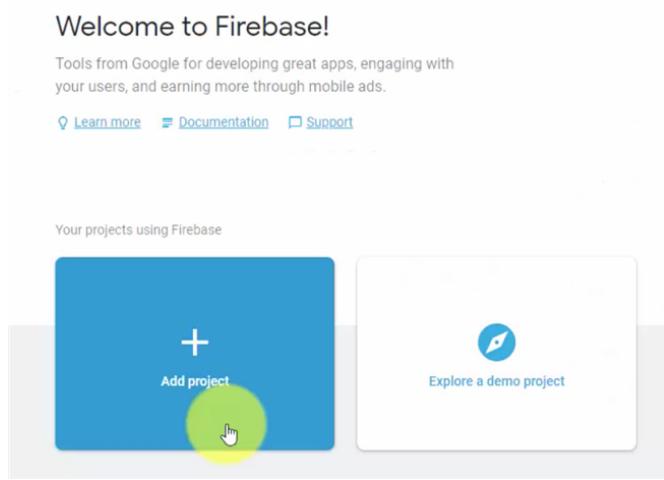
To Firebase enable your application, there are 3 steps you need to follow:

1. Creating a Firebase project and registering your FireMonkey project in the Google Firebase console
2. Creating or updating an existing FireMonkey project in RAD Studio 10.3.1 using the built-in push notification component level support
3. Making modifications to your FireMonkey project to support Firebase instead of Google Cloud

# Creating a Firebase project and registering your FireMonkey project in the Google Firebase console

Once you've downloaded the GetIt package and unzipped the files, follow the steps below.

1. Visit <https://console.firebaseio.google.com/> and click on "+Add Project", then enter a name for your project, i.e. FirebaseApp.



2. You will be asked to review and accept the Google terms before clicking "Create Project".

The screenshot shows the 'Add a project' dialog box. It includes fields for 'Project name' (set to 'Firebal'), 'Project ID' (set to 'fire-494e3'), and 'Locations' (set to 'United States (Analytics)' and 'nam5 (us-central) (Cloud Firestore)'). There is a checked checkbox for 'Use the default settings for sharing Google Analytics for Firebase data' with a list of five sub-options. At the bottom, there is an unchecked checkbox for accepting terms, a 'Cancel' button, and a 'Create project' button.

Add a project

Project name: Firebal

Project ID: fire-494e3

Locations:

- United States (Analytics)
- nam5 (us-central) (Cloud Firestore)

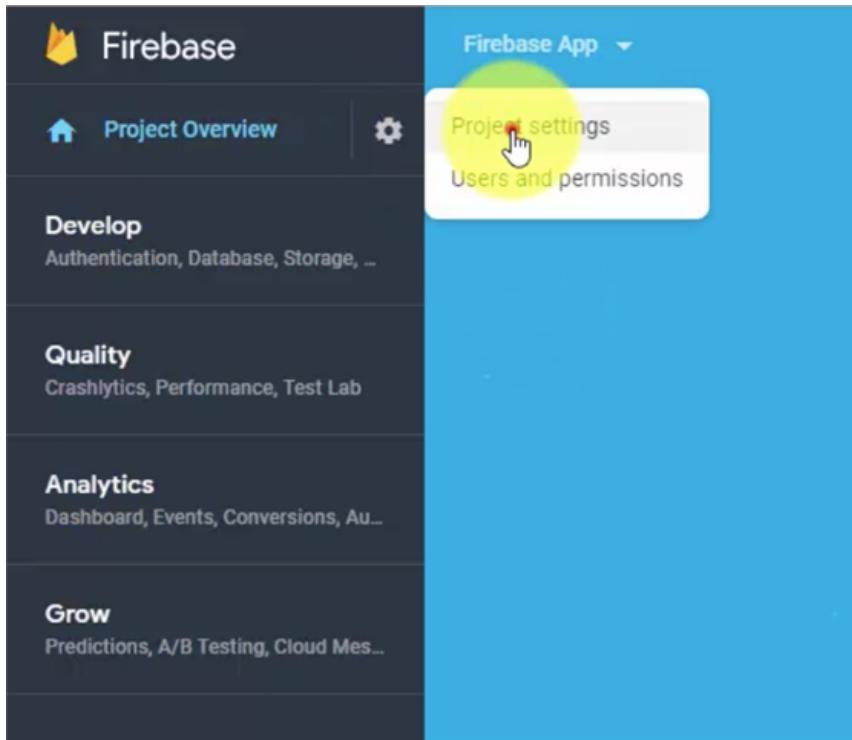
Use the default settings for sharing Google Analytics for Firebase data

- ✓ Share your Analytics data with all Firebase features
- ✓ Share your Analytics data with Google to improve Google Products and Services
- ✓ Share your Analytics data with Google to enable technical support
- ✓ Share your Analytics data with Google to enable Benchmarking
- ✓ Share your Analytics data with Google Account Specialists

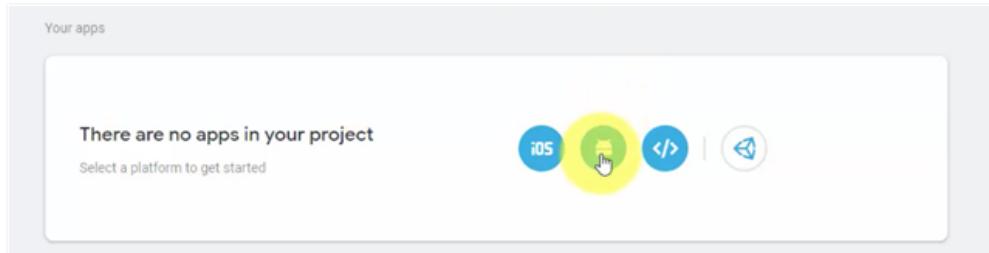
I accept the [controller-controller terms](#). This is required when sharing Analytics data to improve Google Products and Services. [Learn more](#)

Cancel Create project

From the left navigation pane, navigate to the gear icon and select 'Project settings'.



4. Click on the Android icon to access the "Add Firebase to your Android app" page.



5. The default package name for a FireMonkey application is `com.embarcadero.packagename`. In this example, we're changing it to `com.embarcadero.FirebaseApp`, but you should change this to `com.yourcompany.packagename` for your own applications.

Note: For existing applications that are available in the Google Play Store today, you will need to add the package name of that project here instead of assigning a new name. Click "Register app".

## × Add Firebase to your Android app

1 Register app

Android package name

App nickname (optional)

Debug signing certificate SHA-1 (optional)

Required for Dynamic Links, Invites, and Google Sign-In or phone number support in Auth. Edit SHA-1s in Settings.

**Register app**

2 Download config file

3 Add Firebase SDK

4 Run your app to verify installation

## 6. Download the config file

× Add Firebase to your Android app

1 Register app  
Android package name: com.embarcadero.FirebaseApp

2 Download config file

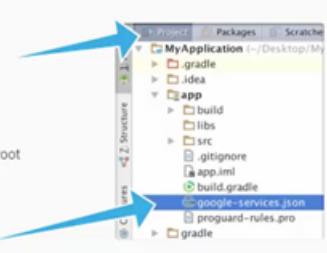
**Download google-services.json**

Switch to the Project view in Android Studio to see your project root directory.

Move the google-services.json file you just downloaded into your Android app module root directory.



Instructions for Android Studio below | C++



3 Add Firebase SDK

4 Run your app to verify installation

Click 'Next' and skip this step at this point.

Manager. This file contains all possible Firebase settings.

While you can update all entries in the template, only two parameters need to be filled in:

google\_app\_id and gcm\_defaultSenderId.

You will find the required values in the google-services.json file which you downloaded in step 6.

```
<resources>
  <string name="gcm_defaultSenderId" translatable="false">428073972981</string> <!-- project_info/project_number -->
  <string name="google_app_id" translatable="false">1:596768615416:android:478ee35511076a5f</string> <!-- (YOUR_CLIENT)/client_info/mobilesdk_app_id -->
```

Any unused parameters should be removed from the xml file at this point. Next a new node should be added in this file:

```
<string name="fcmFallbackNotificationChannelLabel"
translatable="false">Notification channel for Firebase</string>
```

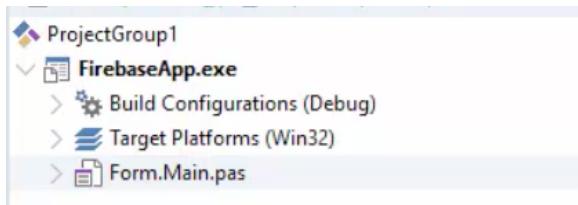
It specifies title for default notification channel. Save this strings.xml file to your FireMonkey application project folder, i.e. C:\MyFireBaseApplication

## Creating or updating an existing FireMonkey project in RAD Studio 10.3.1 using the built-in push notification component level support

8. Create your new FireMonkey application by launching Delphi, C++Builder or RAD Studio 10.3.1 and selecting File > New > Multi-Device Application.

If you have an existing FireMonkey project that you want to update, open it at this point.

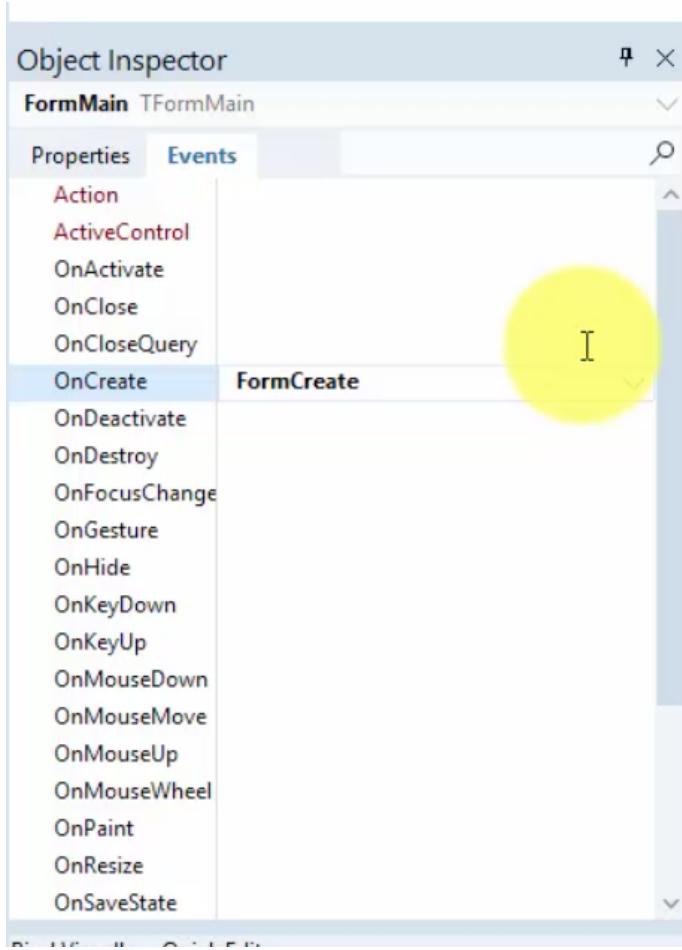
9. Change the FMX project name to match the project name registered in the Firebase console, i.e. FirebaseApp



Save your project files to the same directory as the updated strings.xml file from step 7, i.e.

Project Options matches the package name registered in the Firebase configuration.

10. Add a TMemo control to your form for showing the Firebase event logs. Rename the memo control to MemoLog. Next, we need to initialize the Firebase push notification service and connection using an OnCreate event.



See the Snippets.txt file included in the GetIt package. The Snippets.txt file includes the code for creating the push notification service and connection, and for the event handlers described below. You can copy the code for creating the push notification service and connection from the snippet into your FireMonkey project.

```

FMX.Types, FMX.Controls, FMX.Forms, FMX.Graphics, FMX.Dialogs,
FMX.Controls.Presentation, FMX.ScrollBox, FMX.Memo, System.PushNotification;

10 type
11   TFormMain = class(TForm)
12     MemoLog: TMemo;
13     procedure FormCreate(Sender: TObject);
14   private
15     { Private declarations }
16   public
17     { Public declarations }
18   end;

19 var
20   FormMain: TFormMain;

21 implementation

22 {$R *.fmx}

23 uses
24   FMX.PushNotification.Android;

25 procedure TFormMain.FormCreate(Sender: TObject);
26 var
27   PushService: TPushService;
28   ServiceConnection: TPushServiceConnection;
29 begin
30   PushService := TPushServiceManager.Instance.GetServiceByName(TPushService.TServiceNames.GCM);
31   ServiceConnection := TPushServiceConnection.Create(PushService);
32   ServiceConnection.Active := True;
33   ServiceConnection.OnChange := OnServiceConnectionChange;
34   ServiceConnection.OnReceiveNotification := OnReceiveNotificationEvent;
35
36   FDeviceId := PushService.DeviceIDValue[TPushService.TDeviceIDNames.DeviceId];
37   MemoLog.Lines.Add('DeviceID: ' + FDeviceId);
38   MemoLog.Lines.Add('Ready to receive!');
39 end;
40
41 end.

```

Note: The uses statement for Android push notifications needs to be in an IFDEF if you want to compile the same application for other platforms.

This service instance is used to register a device in Firebase and receive a unique device token that is used as a device service of the push message recipient. For this code to compile, you need to add two fields to your form class:

```
FDeviceId: string;      FDeviceToken: string;
```

As you can see in the included snippet, we assigned two handlers: 1) for receiving push and 2) for getting changes, specifically the receipt of the device token.

```

procedure OnServiceConnectionChange(Sender: TObject; PushChanges:
TPushService.TChanges);
procedure OnReceiveNotificationEvent(Sender: TObject; const
ServiceNotification: TPushServiceNotification);

```

Snippets.txt strings.xml google-services.json

```

1  uses
2    FMX.PushNotification.Android
3  var
4    PushService: TPushService;
5    ServiceConnection: TPushServiceConnection;
6  begin
7    PushService := TPushServiceManager.Instance.GetServiceByName(TPushService.TServiceNames.GCM);
8    ServiceConnection := TPushServiceConnection.Create(PushService);
9    ServiceConnection.Active := True;
10   ServiceConnection.OnChange := OnServiceConnectionChange;
11   ServiceConnection.OnReceiveNotification := OnReceiveNotificationEvent;
12
13   FDeviceId := PushService.DeviceIDValue[TPushService.TDeviceIDNames.DeviceId];
14   MemoLog.Lines.Add('DeviceID: ' + FDeviceId);
15   MemoLog.Lines.Add('Ready to receive!');
16 end;
17
18 procedure TFormMain.OnReceiveNotificationEvent(Sender: TObject; const ServiceNotification: TPushServiceNotification);
19 var
20   MessageText: string;
21 begin
22   MemoLog.Lines.Add('DataKey = ' + ServiceNotification.DataKey);
23   MemoLog.Lines.Add('Json = ' + ServiceNotification.Json.ToString());
24   MemoLog.Lines.Add('DataObject = ' + ServiceNotification.DataObject.ToString());
25 {$IFDEF ANDROID}
26   MessageText := ServiceNotification.DataObject.GetValue('gcm.notification.body').Value;
27 {$ENDIF};
28   MemoLog.Lines.Add(DateTimeToStr(Now) + ' Message = ' + MessageText);
29 end;
30
31 procedure TFormMain.OnServiceConnectionChange(Sender: TObject; PushChanges: TPushService.TChanges);
32 var
33   PushService: TPushService;
34 begin
35   if TPushService.TChange.DeviceToken in PushChanges then
36   begin
37     {$IFDEF ANDROID}
38       PushService := TPushServiceManager.Instance.GetServiceByName(TPushService.TServiceNames.GCM);
39       FDeviceToken := PushService.DeviceTokenValue[TPushService.TDeviceTokenNames.DeviceToken];
40       MemoLog.Lines.Add('FireBase Token: ' + FDeviceToken);
41       Log.d('Firebase device token: token=' + FDeviceToken);
42     {$ENDIF}
43   end;
44 end;
45
46
47
48
49
50

```

```
begin
  PushService := TPushServiceManager.Instance.GetServiceByName(TPushService.TServiceNames.GCM);
  ServiceConnection := TPushServiceConnection.Create(PushService);
  ServiceConnection.Active := True;
  ServiceConnection.OnChange := OnServiceConnectionChange;
  ServiceConnection.OnReceiveNotification := OnReceiveNotificationEvent;

  FDeviceId := PushService.DeviceIDValue[TPushService.TDeviceIDNames.DeviceId];
  MemoLog.Lines.Add('DeviceID: ' + FDeviceId);
  MemoLog.Lines.Add('Ready to receive!');
end;

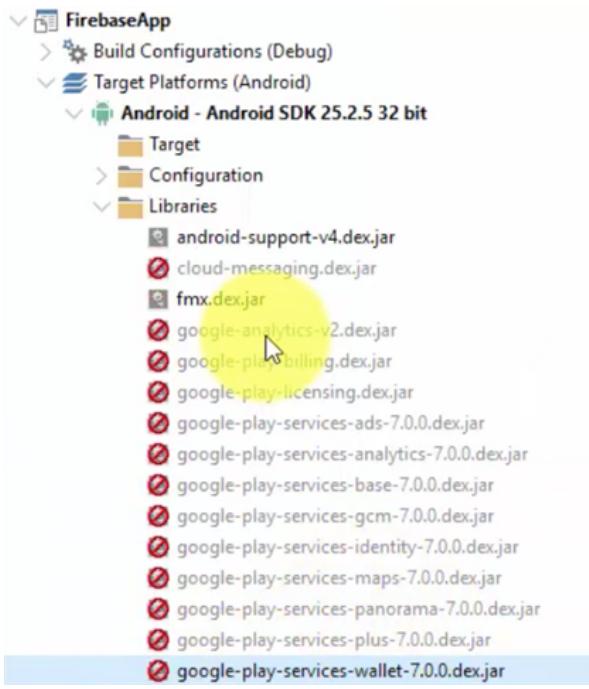
procedure TFormMain.OnReceiveNotificationEvent(Sender: TObject; const ServiceNotification: TPushServiceNotification);
var
  MessageText: string;
begin
  MemoLog.Lines.Add('DataKey = ' + ServiceNotification.DataKey);
  MemoLog.Lines.Add('Json = ' + ServiceNotification.Json.ToString);
  MemoLog.Lines.Add('DataObject = ' + ServiceNotification.DataObject.ToString);
{$IFDEF ANDROID}
  MessageText := ServiceNotification.DataObject.GetValue('gcm.notification.body').Value;
{$ENDIF};
  MemoLog.Lines.Add(DateTimeToStr(Now) + ' Message = ' + MessageText);
end;

procedure TFormMain.OnServiceConnectionChange(Sender: TObject;
  PushChanges: TPushService.TChanges);
var
  PushService: TPushService;
begin
  if TPushService.TChange.DeviceToken in PushChanges then
  begin
    {$IFDEF ANDROID}
    PushService := TPushServiceManager.Instance.GetServiceByName(TPushService.TServiceNames.GCM);
    FDeviceToken := PushService.DeviceTokenValue[TPushService.TDeviceTokenNames.DeviceToken];
    MemoLog.Lines.Add('FireBase Token: ' + FDeviceToken);
    Log.d('Firebase device token: token=' + FDeviceToken);
    {$ENDIF}
  end;
end;
end.
```

We'll print the token into the device log for testing.

## Making modifications to your FireMonkey project to support Firebase instead of Google Cloud Messaging

11. In the Project Manager on the right, expand Android > Libraries and disable all Google Play and Cloud Messaging Libraries by manually right-clicking on each of them



12. Next, right-click on Libraries to add new Firebase libraries and updated Google Play Services libraries. Those can be found in the Firebase - FMX archive that you downloaded from the GetIt Package Manager (Firebase - FMX > Supporting Files > jars and Firebase - FMX > Supporting Files > jars > Google Play Services). Notice you need to add all of the JAR files in those two folders.

13. Next, add the updated FMX.PushNotification.Android.pas with Firebase support and AndroidAPI.JNI.Firebase.pas file to your project. These files can be found in the Firebase - FMX archive that you downloaded from the GetIt Package Manager (Firebase - FMX > Supporting Files). To add both files to your project, right-click on FirebaseApp in the Project Manager and select "Add".

Save the project.

14. Navigate to Project > Deployment Manager and add the strings.xml file from your FireMonkey application project folder, i.e. C:\MyFireBaseApplication (see step 7).

Update the Remote Path to res\values\ for the strings.xml file and Save the changes.

<input checked="" type="checkbox"/>	\$(BDS)\bin\Artwork\A...	FM_LauncherIcon_144x1...	Android_Launch...	Debug	[Android]	res\drawable-xhdpi\	ic_launcher.png	Not Connected	Always
<input checked="" type="checkbox"/>	\$(BDS)\bin\Artwork\A...	FM_SplashImage_470x32...	Android_Splash...	Debug	[Android]	res\drawable-normal\	splash_image.png	Not Connected	Always
<input checked="" type="checkbox"/>	\$(BDS)\bin\Artwork\Android\	LauncherIcon_96x96...	Android_Launch...	Debug	[Android]	res\drawable-xhdpi\	ic_launcher.png	Not Connected	Always
<input checked="" type="checkbox"/>		strings.xml	File	Debug	[Android]	res\values\	strings.xml	Not Connected	Always
<input checked="" type="checkbox"/>	Android\Debug\	styles-v21.xml	AndroidSplashS...	Debug	[Android]	res\values-v21\	styles.xml	Not Connected	Always

15. The last step involves changes to the Android manifest template file (AndroidManifest.template.xml). You can find the Android manifest file in your FireMonkey application

Open the xml file and the Snippets.txt file provided in the GetIt package. You can copy and paste the snippet code into your manifest file, after the section

```
<%activity%>
```

```
<%receivers%>
```

and before

```
</application>
```

```
</manifest>
```

When done, hit Save.

```

32      <meta-data android:name="android.app.lib_name"
33          android:value="@libNameValue" />
34      <intent-filter>
35          <action android:name="android.intent.action.MAIN" />
36          <category android:name="android.intent.category.LAUNCHER" />
37      </intent-filter>
38  </activity>
39  <%activity%>
40  <%receivers%>
41
42  <service
43      android:name="com.embarcadero.firebaseio.ProxyFirebaseMessagingService"
44      android:permission="com.google.android.c2dm.permission.SEND"
45      android:exported="false">
46      <intent-filter>
47          <action android:name="com.google.firebaseio.MESSAGING_EVENT" />
48      </intent-filter>
49  </service>
50
51  <!--
52      FirebaseMessagingService performs security checks at runtime,
53      but set to not exported to explicitly avoid allowing another app to call it.
54  -->
55  <service
56      android:name="com.google.firebaseio.messaging.FirebaseMessagingService"
57      android:exported="true" >
58      <intent-filter android:priority="-500" >
59          <action android:name="com.google.firebaseio.MESSAGING_EVENT" />
60      </intent-filter>
61  </service>
62
63  <service
64      android:name="com.google.firebaseio.components.ComponentDiscoveryService"
65      android:exported="false" >
66      <meta-data
67          android:name="com.google.firebaseio.components:com.google.firebaseio.iid.Registrar"
68          android:value="com.google.firebaseio.components.ComponentRegistrar" />
69  </service>
70
71  <!--
72      FirebaseInstanceIdService performs security checks at runtime,
73      no need for explicit permissions despite exported="true"
74  -->
75  <service
76      android:name="com.google.firebaseio.iid.FirebaseInstanceIdService"
77      android:exported="true" >
78      <intent-filter android:priority="-500" >
79          <action android:name="com.google.firebaseio.INSTANCE_ID_EVENT" />
80      </intent-filter>
81  </service>
82
83  <receiver
84      android:name="com.google.firebaseio.iid.FirebaseInstanceIdReceiver"
85      android:exported="true" >

```

16. Deploy your FMX application to your Android device. You should now see the device token in the log and that the app was automatically registered in FireBase.



To send a push message to the device, we will need to copy the Firebase token. You can copy it from the memo, or as an alternative, you can take it from the adb logcat. See the Log.d method call in the code snippet.

17. Navigate to [console.firebaseio.google.com](https://console.firebaseio.google.com) in the browser and select the project you previously created. Then select Grow > Cloud Messaging > Send your first message from the side menu.
18. Compose your notification by entering a title and message, then click send test message. Next, enter the device token you previously copied and click Test. This will push the message to our test device.

1 Notification

Notification title (only for Android and watchOS) ⓘ

Notification label (optional) ⓘ

Notification text

Send test messageNext

2 Target

3 Scheduling

Send now

4 Conversion events (optional)

5 Additional options (optional)

Save as draftReview

## Test on device

You can test this campaign by entering or selecting the [FCM registration tokens](#) of your development device below.

Add an FCM registration token

e8MLrzob76g:APA91bGCFqtcuK7XciobZWwnNWi\_-ytVkJ-SKEs1-zdG\_BwViR...

Cancel

Test



100 % 20:57

DeviceID: ED9C99181A6D5679B4D572F0C1628

Ready to receive!

FireBase Token: e8MLrzob76g:APA91bGCFqtcuI

DataKey = gcm

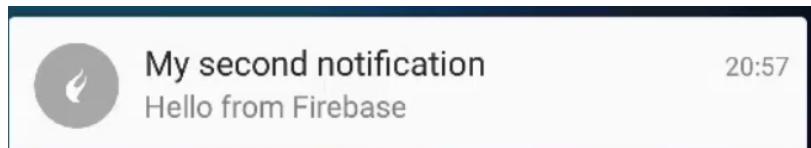
Json = {"google.c.a.udt": "0", "google.delivered\_pri

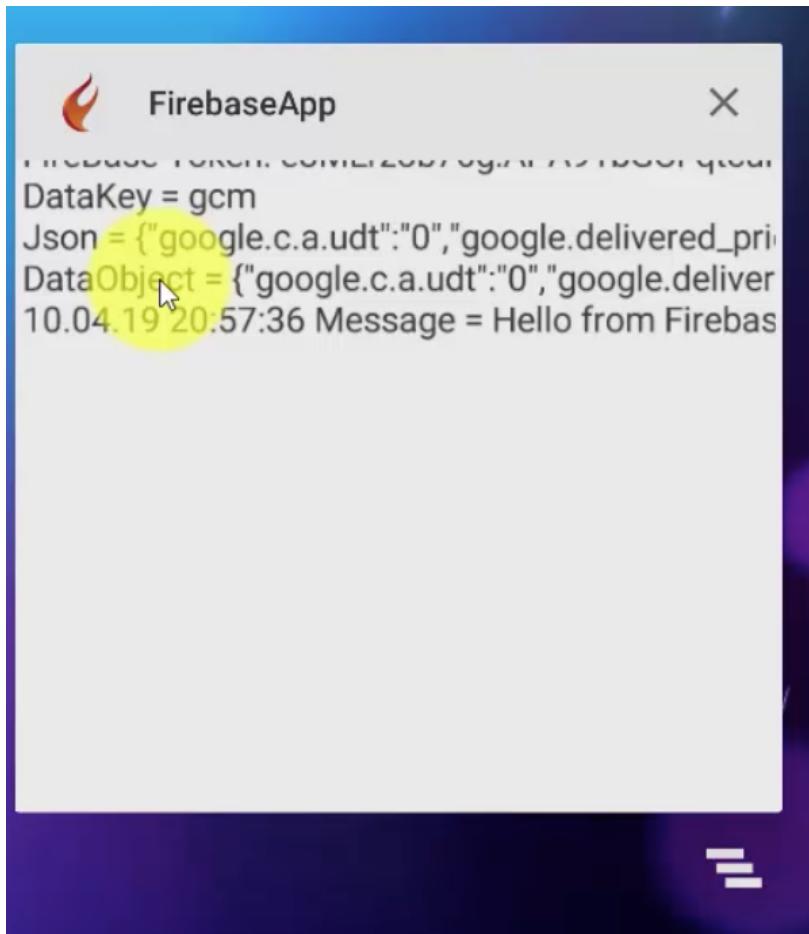
DataObject = {"google.c.a.udt": "0", "google.deliver

10.04.19 20:57:36 Message = Hello from Firebas



As you can see in the screenshot above, the log shows that the message was successfully received. Messages also appear in the Android notification center.





*PS. There is a caveat applying this patch: The TMapView component will stop working because the process updates Google Play jars, and they are not fully compatible. We plan addressing this in coming updates.*

9 comments 0 members are here



Offline [Chris Chuah](#) 4 months ago

How about for iOS? I have been trying for past few months trying to get it to work for iOS but nothing seems to work.



[Marco Cantu](#) 4 months ago in reply to [Chris Chuah](#)

There was an open QP related with iOS, I think it was fixed for 10.3.1, but need to check. In terms of offering integration out-of-the-box for Firebase on iOS, this is under consideration for 10.4



Offline [FunChan](#) 4 months ago

strings.xml : Look at the pictures and explanations and make them yourself.

Snippets.txt C:\Users\Public\Documents\Embarcadero\Studio\20.0\CatalogRepository\AndroidPushNotificationsPatch-1.0\FireBase



Offline [Jose Morango](#) 4 months ago



Offline [Juan Martinez](#) 3 months ago

Hello



Offline [Juan Martinez](#) 3 months ago

The onreceiveonnotificationevent never raised if app is in background (or not running). I received message in Android Notification Center, but message only if app is running and not in background.

Could you help me please?



Offline [Janderson Gomes H10267](#) 2 months ago

I tried to do the above procedure and could not make the application work.

After many attempts, I started a project from scratch, just ticked the Project -> Options -> Entitlement List -> Receive push notification screen, compiled for android, and deployed to mobile, and the application doesn't open.

If I uncheck the option, the application opens.

The app has no buttons, no commands.

I am using delphi 10.3.2 RIO

Thank you.

Janderson Henrique



[Marco Cantu](#) 2 months ago in reply to [Janderson Gomes H10267](#)

You need also to provide the specific information for push notification (ID, etc) as registered in firebase console.



Offline [Janderson Gomes H10267](#) 2 months ago in reply to [Marco Cantu](#)

Thanks, I did it and resolved, the app opens, I get the notification within the app.

But when I close the application the notification is not enough