



Flutter Bootcamp

Firestore & Plugins

Emanuel López – emanuel.lopez@globant.com

Repaso

- Navigator 1.0
- Named Routes
- Navigator 2.0
- GoRouter
- build_runner
- Librerías o paquetes
 - `flutter_secure_storage`
 - `flutter_native_splash`
 - `url_launcher`
 - `google_fonts`
 - `shimmer`

Bases de datos NoSQL

Una base de datos NoSQL es una solución ideal para aplicaciones modernas que requieren **flexibilidad, escalabilidad y manejo de grandes volúmenes de datos**.

Son adecuadas cuando los datos a guardar son no estructurados o semi estructurados, o cuando pueden cambiar frecuentemente sin necesidad de alterar el esquema (como redes sociales, big data, o aplicaciones de IoT).

Firestore es un ejemplo de base de datos NoSQL basada en documentos. **Organiza los datos en colecciones y documentos en lugar de tablas y filas**, lo que permite flexibilidad en la estructura de datos.

Los datos se actualizan automáticamente en todos los dispositivos conectados a medida que se producen cambios.

Cloud Firestore

Instalación de dependencias:

```
flutter pub add firebase_core cloud_firestore
```

Inicialización:

```
void main() async {  
  WidgetsFlutterBinding.ensureInitialized();  
  await Firebase.initializeApp();  
}
```

Acceso a la base de datos:

```
final _firestoreDatabase = FirebaseFirestore.instance;
```

Cloud Firestore

Listado de una colección:

```
final snapshot = await _firestoreDatabase
    .collection('contacts')
    .get();
```

Recupero de un documento:

```
final snapshot = await _firestoreDatabase
    .collection('contacts')
    .doc(contactId)
    .get();
```

Cloud Firestore

Inserción con id específico:

```
await _firestoreDatabase
    .collection('contacts')
    .doc(contact.id.toString())
    .set(contact.toMap());
```

Inserción con id autogenerado:

```
await _firestoreDatabase
    .collection('contacts')
    .add(contact.toMap());
```

Cloud Firestore

Actualización:

```
await _firestoreDatabase
    .collection('contacts')
    .doc(contact.id.toString())
    .update(contact.toMap());
```

Borrado:

```
await _firestoreDatabase
    .collection('contacts')
    .doc(contact.id.toString())
    .delete();
```

Cloud Firestore

Permite tener sincronización en tiempo real. Los cambios en documentos o colecciones se propagan en tiempo real a todos los clientes que estén escuchando.

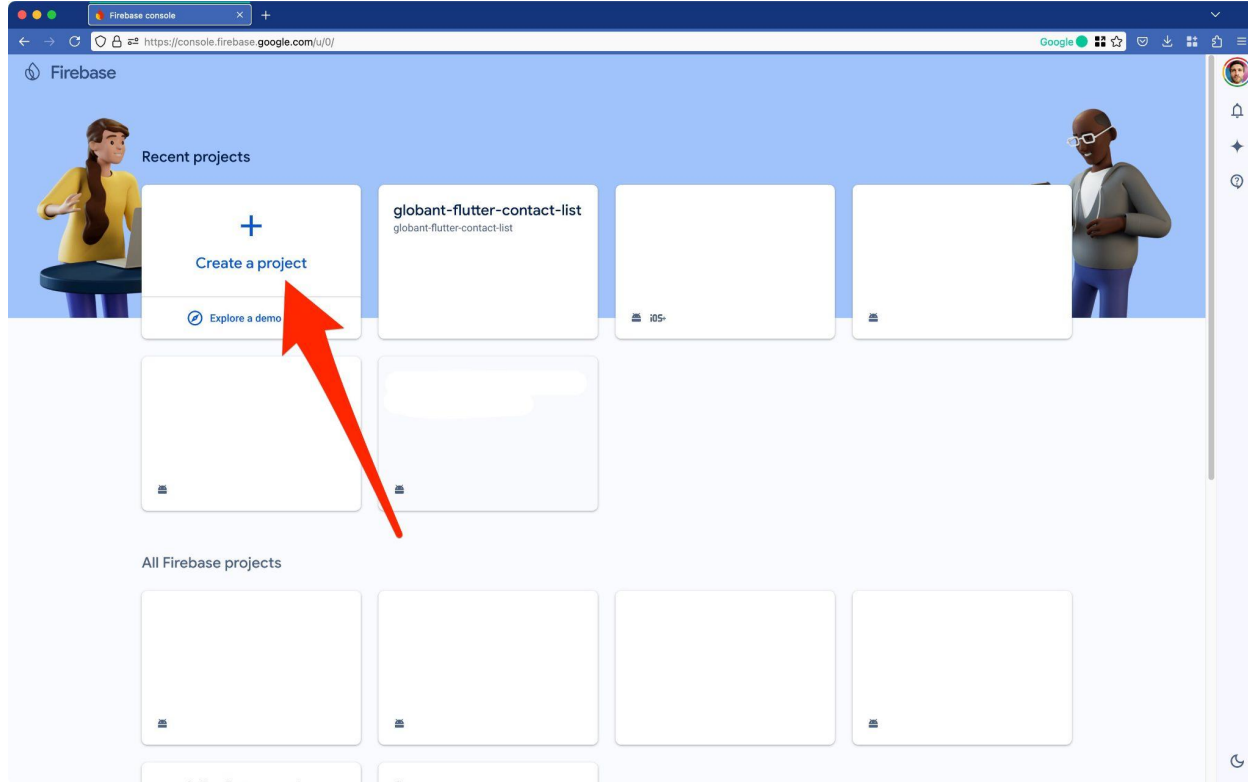
Suscripción a actualizaciones de una colección:

```
_firebaseDatabase.collection('polls').snapshots()  
  .listen((snapshot) => _parseCollection(snapshot));
```

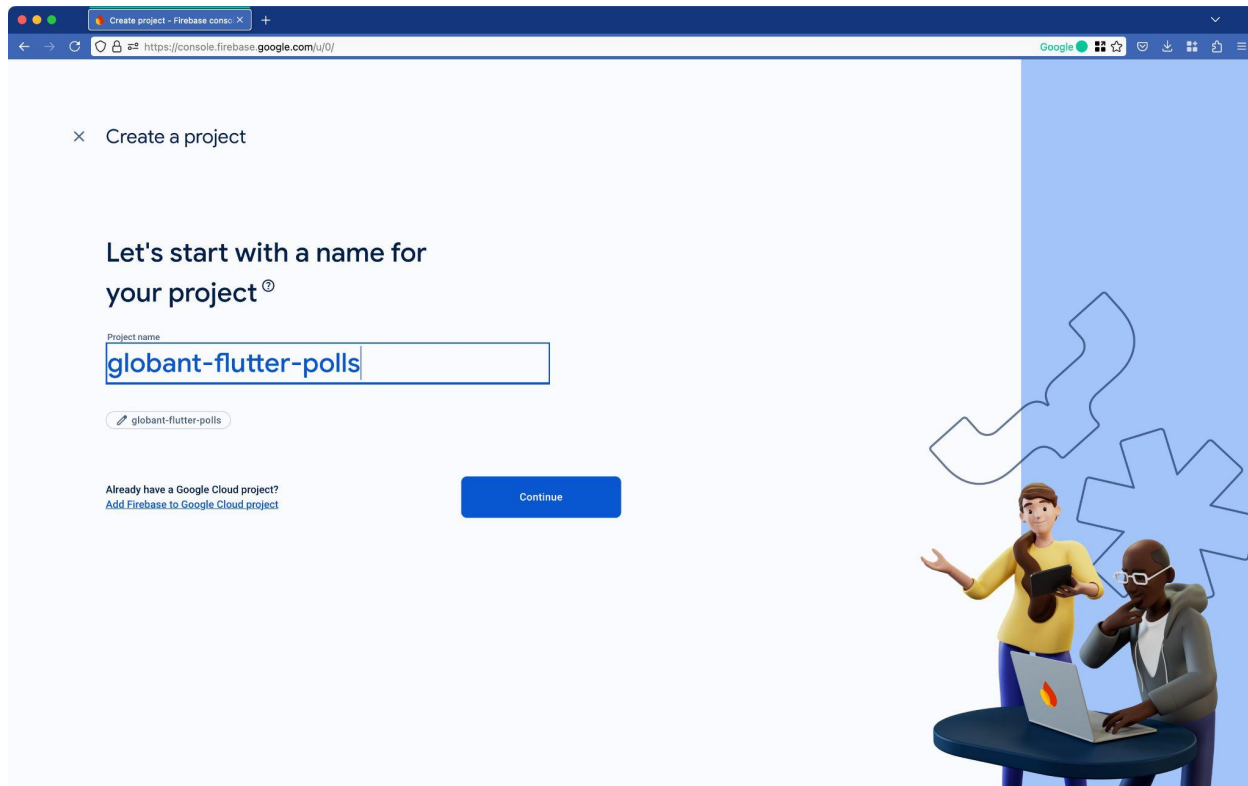
Suscripción a actualizaciones de un documento:

```
_firebaseDatabase  
  .collection('admin')  
  .doc('settings')  
  .snapshots()  
  .listen((document) => _parseDocument(document));
```

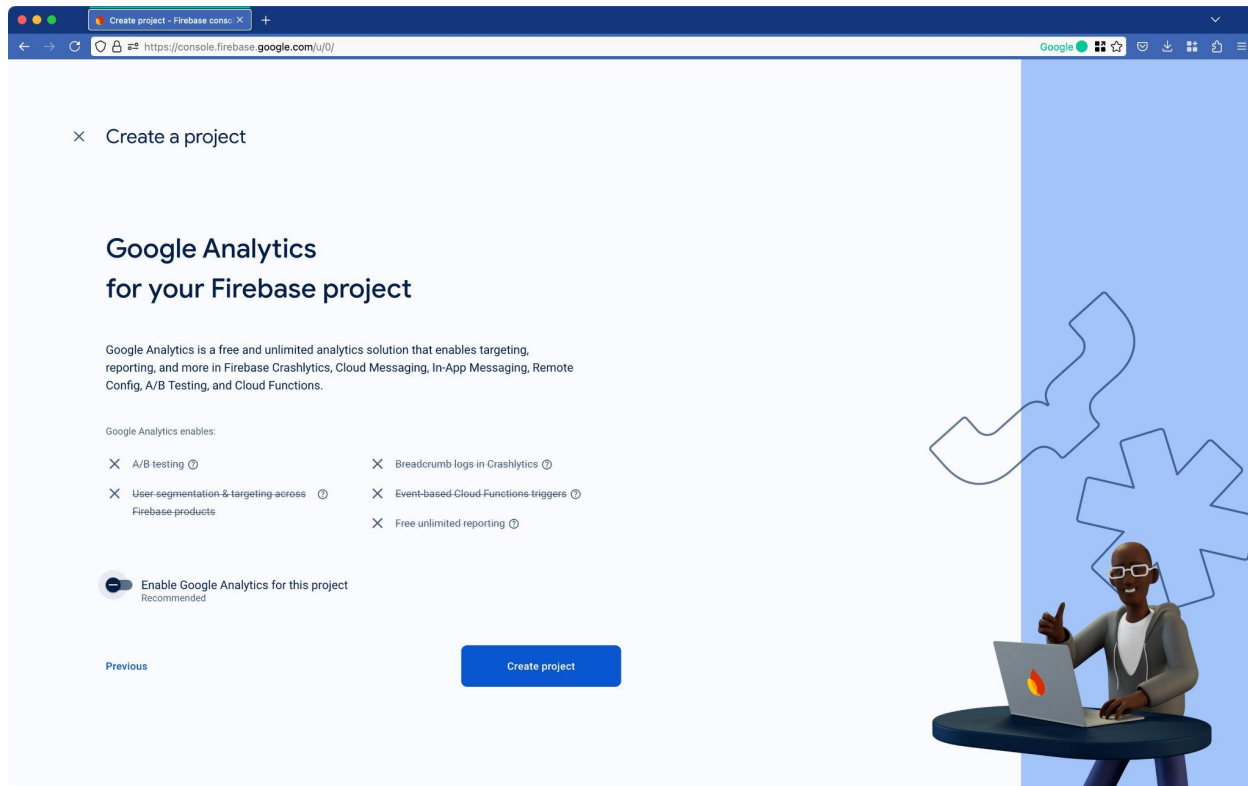

Cloud Firestore



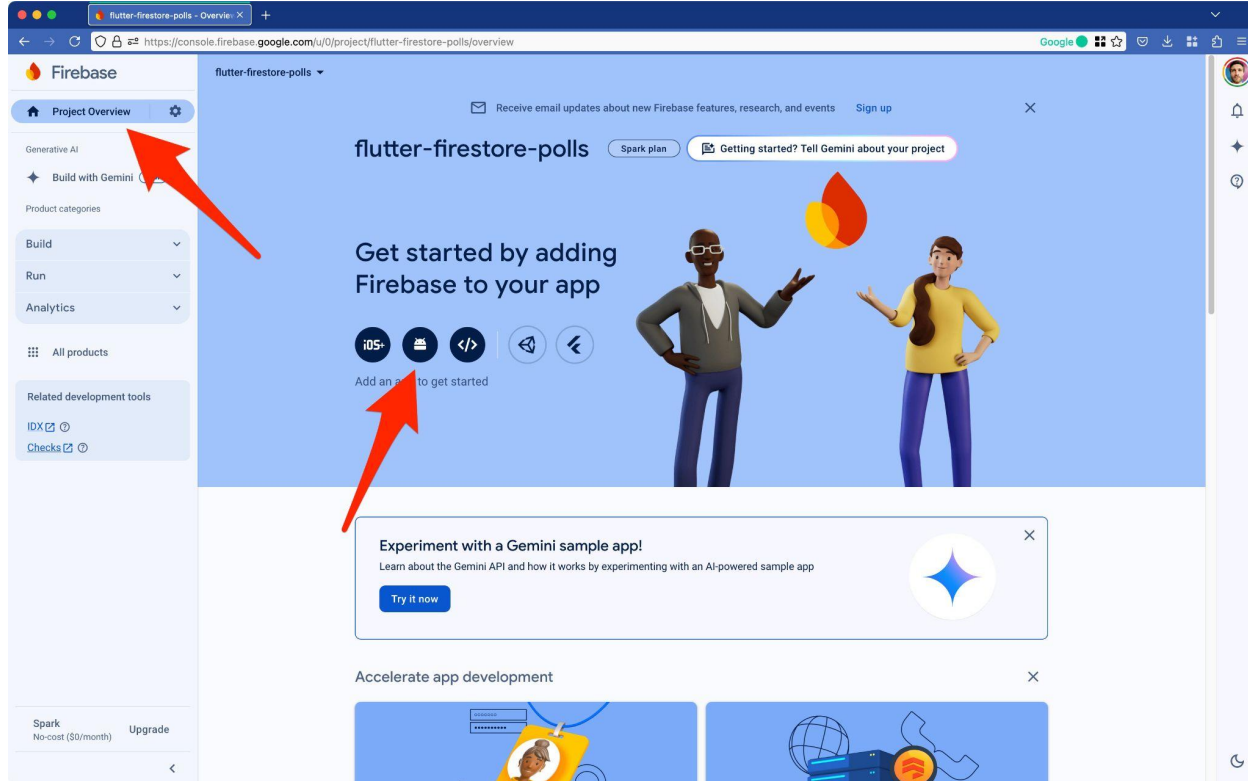
Cloud Firestore



Cloud Firestore



Cloud Firestore



Cloud Firestore

[illegible]


Cloud Firestore


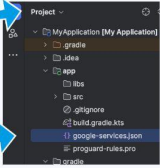
flutter-firestore-polls - Add app: x

https://console.firebase.google.com/u/0/project/flutter-firestore-polls/overview

Go to docs

✕ Add Firebase to your Android app

- ✓ Register app
Android package name: com.example.firestore_polls
- 2 Download and then add config file
Instructions for Android Studio below | [Unity](#) | [C++](#)
[Download google-services.json](#)
Switch to the Project view in Android Studio to see your project root directory.
Move your downloaded google-services.json file into your module (app-level) root directory.

[Next](#)
- 3 Add Firebase SDK
- 4 Next steps



Flutter plugins

Los plugins de Flutter son paquetes que permiten a los desarrolladores acceder a funcionalidades nativas de las plataformas móviles o de escritorio desde código Dart.

Los plugins actúan como un "puente" entre el código Flutter y las APIs nativas, permitiendo que nuestra Flutter aproveche características específicas de cada plataforma, como sensores, cámaras, almacenamiento, servicios de ubicación, etc.

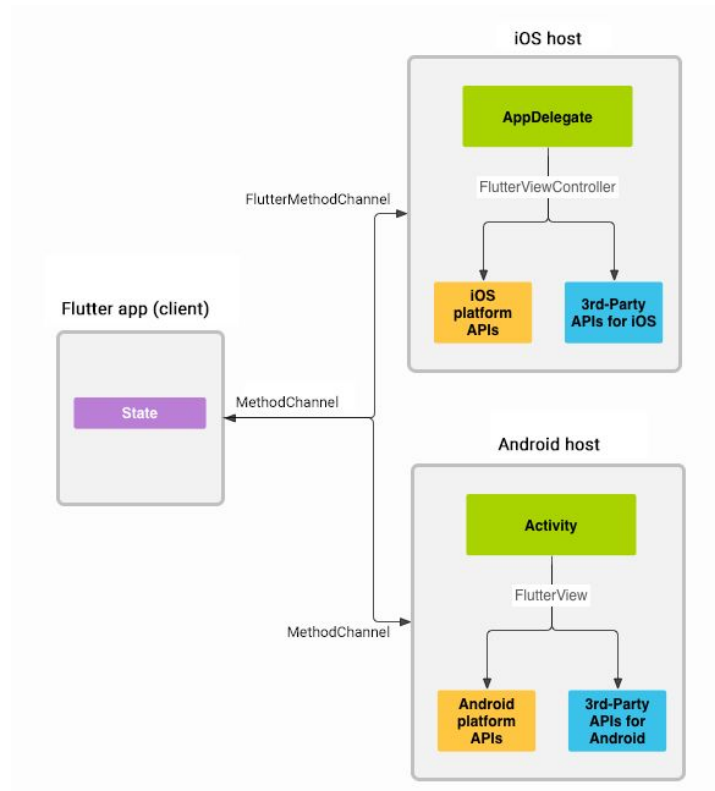
Componentes de un Plugin de Flutter:

- **Código Dart:** Proporciona la API que se utiliza en la aplicación Flutter. Es el punto de entrada para el desarrollador y lo que llamas desde tu código en Dart.
- **Código nativo:** Android (Kotlin o Java), iOS (Swift o Objective-C), Windows (C++), macOS (Objective-C/Swift), Linux (C++).
- **Platform Channel:** Mecanismo que permite la comunicación entre el framework y las plataformas nativas.

Flutter plugins

Como crear un plugin en Flutter:

```
flutter create --template=plugin  
--platforms=android  
android_plugin
```



Platform Channels: MethodChannel

Los Platform Channels son la clave del funcionamiento de los plugins de Flutter. Los **MethodChannel** se utilizan para la comunicación bidireccional (llamar a un método y recibir una respuesta).

1. **Código Dart envía un mensaje:** En nuestra aplicación Flutter, usas el canal de plataforma para enviar un mensaje desde Dart a la plataforma nativa, solicitando una funcionalidad específica.
2. **Código nativo recibe el mensaje y procesa la solicitud:** La plataforma nativa recibe el mensaje a través del mismo canal. Aquí entra en juego el código nativo que ejecuta la funcionalidad solicitada y devuelve un resultado.
3. **El resultado se envía de vuelta a Dart:** Una vez completada la operación en el código nativo, el resultado se devuelve a través del mismo canal, y el código Dart recibe la respuesta.

Platform Channels: EventChannel

Los **EventChannel** están diseñados específicamente para transmitir flujos de datos o eventos continuos desde la plataforma nativa a Flutter.

1. **Flutter:** Se suscribe al **EventChannel** y escucha eventos transmitidos.
2. **Código nativo:** Detecta eventos (por ejemplo, el acelerómetro) y envía esos eventos al canal.
3. **Flutter:** Recibe los eventos y puede actuar en consecuencia (por ejemplo, actualizando la interfaz de usuario).

Links adicionales

- Documentación Firebase:
firebase.google.com/docs/flutter/setup?platform=android
- Flutter Firebase codelabs: docs.flutter.dev/codelabs/flutter-and-firebase
- Writing custom platform-specific code:
docs.flutter.dev/platform-integration/platform-channels
- Flutter plugin for Android and iOS. Get battery level:
appencekit.dev/blog/flutter-create-plugin

Preguntas?



Gracias!

