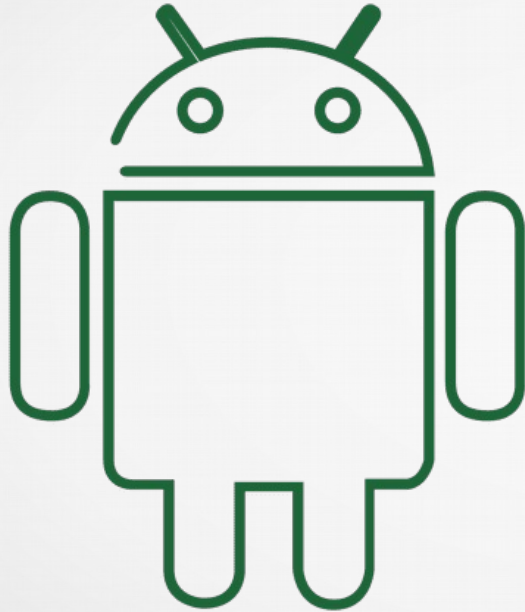


Haciendo una app para probar Cloud Firestore



<https://github.com/dbetm/Workshop-Firestore>

Ruta de implementación recomendada

- 1) Integra los SDK de Cloud Firestore
- 2) Protege los datos
- 3) Agrega datos
- 4) Obtén datos




App alumnos


1) Crea un proyecto de Cloud Firestore


1. Abre **Firebase console** y crea un proyecto nuevo.
2. En la sección **Base de datos**, haz clic en **Probar Firestore Beta**.
3. Haz clic en **Habilitar**.


1) Crea un proyecto de Cloud Firestore


Desarrolla


 Authentication

 Database

 Storage

 Hosting

 Functions

 ML Kit

Calidad

Beta

Cloud Firestore

La próxima generación de Realtime Database por escalado automático y con

Crear base de datos

Reglas de seguridad de Cloud Firestore

Una vez que defines tu estructura de datos, deberás crear reglas para proteger los datos.
[Más información](#)


☐ Comenzar en modo bloqueado

Rechaza todas las operaciones de lectura y escritura para que tu base de datos sea privada

☒ Comenzar en modo de prueba

Permite todas las operaciones de lectura y escritura en tu base de datos para realizar una configuración rápida

```
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write;
    }
  }
}
```

 Cualquier persona que tenga la referencia de tu base de datos podrá realizar operaciones de lectura o escritura en ella

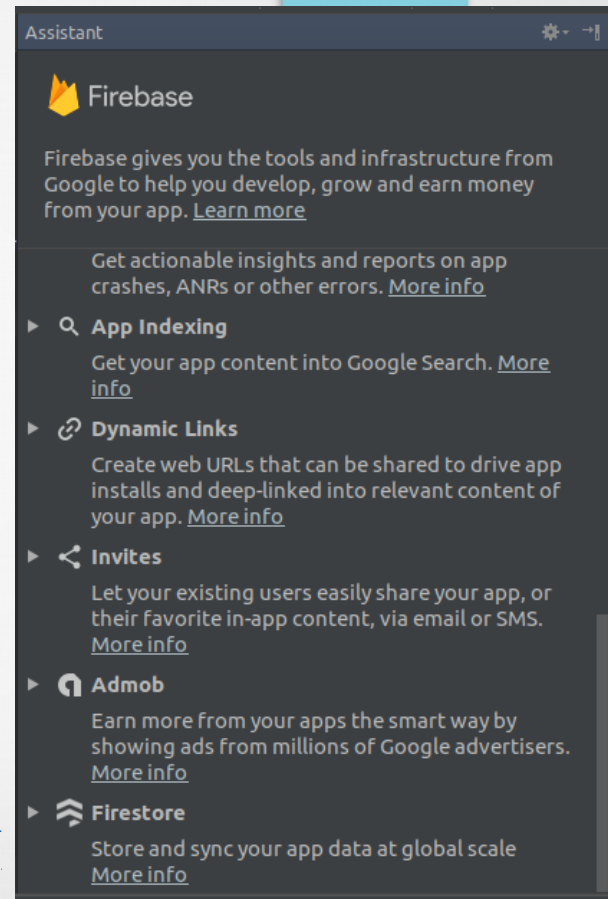
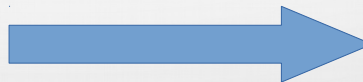
Si habilitas Cloud Firestore Beta no podrás usar Cloud Datastore en este proyecto, especialmente desde la app de App Engine asociada.

Cancelar

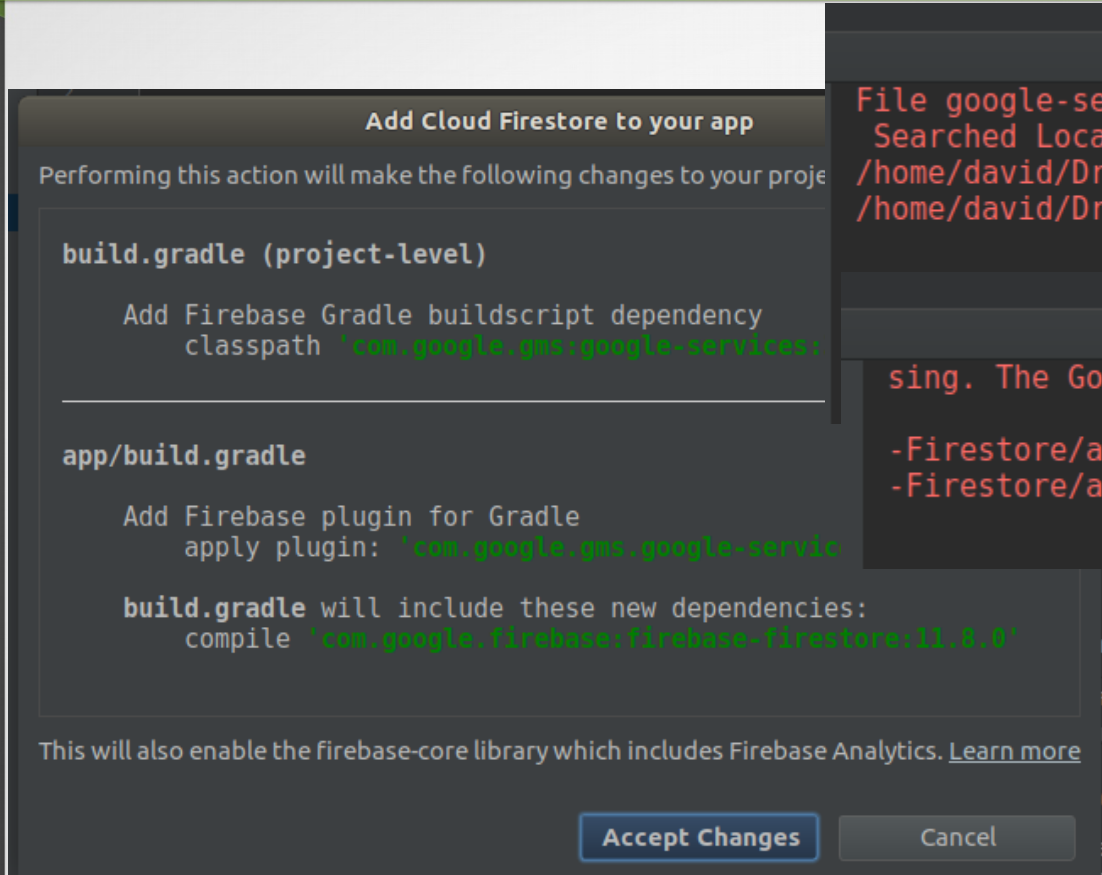
Habilitar

2) Configura tu entorno de programación

- (1) Connect your app to Firebase
- (2) Add Cloud Firestore to your app



2) Configura tu entorno de programación



File google-services.json is missing. The Google Services Plugin
Searched Location:
/home/david/Dropbox/Dev/Workshop-Firestore/app-alumnos/app/src/
/home/david/Dropbox/Dev/Workshop-Firestore/app-alumnos/app/google-services.json

File google-services.json is missing. The Google Services Plugin cannot function without it.
-Firestore/app-alumnos/app/src/debug/google-services.json
-Firestore/app-alumnos/app/google-services.json

3) Inicializa Cloud Firestore

```
public class MainActivity extends AppCompatActivity {  
    // Atributos  
    FirebaseFirestore db;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
        setup();  
    }  
  
    public void setup() {  
        db = FirebaseFirestore.getInstance();  
    }  
}
```

Al correr:

**FirebaseInstanceId:
topic sync
succeeded**

3-4-)→ El modelo de datos

Documento:

nombre: "David"

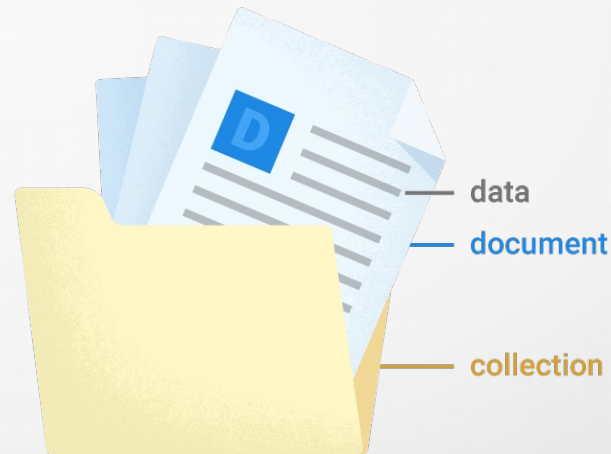
apellido_s:

"Betancourt Montellano"

nombre_usuario: "dbetm"

nacimiento: 1998

Colección: "alumnos"



#Referencia

4) Agrega datos (agregar doc)

app-alumnos

Agregar datos

```
public void agregarDatos(View v) {  
    // Creamos un nuevo alumno con la estructura de datos Map  
    Map<String, Object> alumno = new HashMap<>();  
    alumno.put("nombre", "David");  
    alumno.put("apellido_s", "Betancourt Montellano");  
    alumno.put("nombre_usuario", "dbetm");  
    alumno.put("nacimiento", 1998);  
}
```

Documento agregado con ID:
qjEaSZ3dR8rYhiag32zz

4) Agrega datos (agregar documento).

```
// Referencia a la colección de alumnos
CollectionReference alumnosCollectionRef = db.collection("alumnos");

// Se agrega el documento con un ID generado automáticamente por Cloud Firestore.
alumnosCollectionRef
    .add(alumno)
    .addOnSuccessListener(new OnSuccessListener<DocumentReference>() {
        @Override
        public void onSuccess(DocumentReference documentReference) {
            Toast.makeText(getApplicationContext(),
                "Documento agregado con ID: " + documentReference.getId(),
                Toast.LENGTH_LONG).show();
        }
    })
    .addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            Toast.makeText(getApplicationContext(),
                "Error al tratar de agregar documento: " + e,
                Toast.LENGTH_LONG).show();
        }
    });
}
```

4) Leer datos (obtener documento).

- 1) Copiar el id del documento desde la consola de Firebase.
- 2) Crear nueva actividad (vacía – empty) llamada **DatosAlumno**.
 - 1) Hacer diseño XML en **activity_datos_alumno.xml**
 - 2) Agregar botón que llame el método del siguiente punto en MainActivity
- 3) Implementar método ***recuperarDocumento(View v)*** en el MainActivity.
- 4) En el método **onCreate()**:
 - 1) Referenciar los TextView
 - 2) Recuperar parámetro que viene de **MainActivity**.
 - 3) Agregar los datos recuperados del String en una estructura de datos.
 - 4) Se setean los TextView.
- 5) Lanzar la app y clic en el botón de “Recuperar documento”.

4) Leer datos (obtener documento).

* Es necesario tener el id del documento que se desea obtener, puedes obtenerlo de la consola de Firebase.

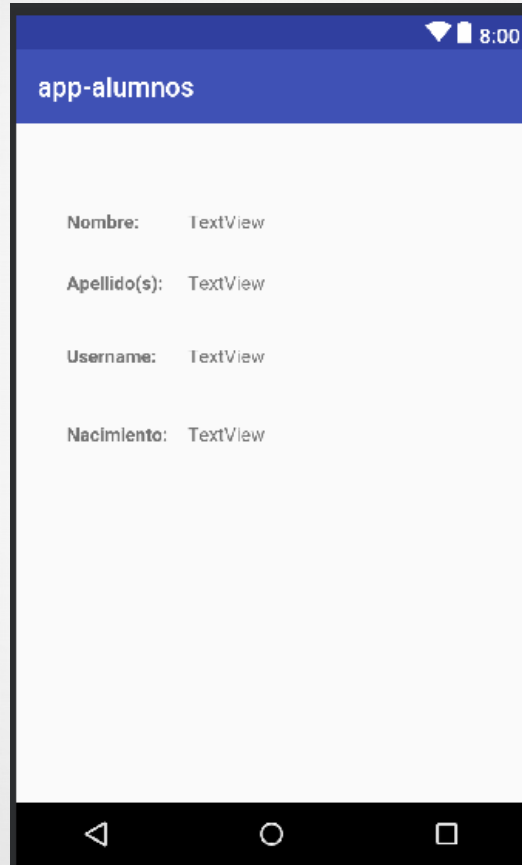
The screenshot shows the Firebase console interface. At the top, the breadcrumb navigation is 'home > alumnos > Tuj7ozP3n6Mm...'. Below this, there are three main sections:

- Left Panel (Database Structure):** Shows the hierarchy 'movilesdbm' > 'alumnos'. The 'alumnos' collection is expanded, showing a document with ID 'Tuj7ozP3n6MmVHRPE4DG'.
- Middle Panel (Document Viewer):** Displays the content of the selected document. It shows a key 'qjEaSZ3dR8rYhiag32zz'.
- Right Panel (Document Details):** Shows the JSON representation of the document data:

```
{  "apellido_s": "Betancourt Montellano",  "nacimiento": 1998,  "nombre": "David",  "nombre_usuario": "dbetm"}
```


4) Leer datos (obtener documento).

Diseños XML



4) Leer datos (obtener documento).

```
public void recuperarDocumento(View v) {  
    // Declaramos e inicializamos la referencia del documento  
    DocumentReference docRef = db.collection("alumnos").document("Tuj7ozP3n6MmVHRPE4DG");  
    docRef.get().addOnCompleteListener(new OnCompleteListener<DocumentSnapshot>() {  
        @Override  
        public void onComplete(@NonNull Task<DocumentSnapshot> task) {  
            if(task.isSuccessful()) {  
                DocumentSnapshot alumno = task.getResult();  
                if(alumno.exists()) { // Pregunta existencial (literal)  
                    Intent intent = new Intent(getApplicationContext(), DatosAlumno.class);  
                    // alumno.getData(); Retorna un Map<String, Object>  
                    intent.putExtra("datosAlumno", alumno.getData().toString());  
                    startActivity(intent);  
                }  
                else {  
                    Toast.makeText(getApplicationContext(), "¡Alumno no encontrado!",  
                        Toast.LENGTH_LONG).show();  
                }  
            }  
            else {  
                Toast.makeText(getApplicationContext(), "get failed with " + task.getException(),  
                    Toast.LENGTH_LONG).show();  
            }  
        }  
    });  
}
```

4) Leer datos (obtener documento).

```
protected void onCreate(Bundle savedInstanceState) {  
    // ...  
    // Referencia a los TextView's  
    TextView txtNombre = findViewById(R.id.txtNombre);  
    TextView txtApellido_s = findViewById(R.id.txtApellido);  
    TextView txtUsername = findViewById(R.id.txtUsername);  
    TextView txtNacimiento = findViewById(R.id.txtNacimiento);  
  
    String datosAlumno = getIntent().getExtras().getString("datosAlumno");  
    datosAlumno = datosAlumno.replace("{", "");  
    datosAlumno = datosAlumno.replace("}", "");  
    datosAlumno = datosAlumno.replace(" ", ""); // Quitar espacios en blanco  
  
    HashMap<String, Object> datos = new HashMap<>();  
    String []pares = datosAlumno.split(",");  
  
    // Los agregamos a la estructura de datos  
    for (int i = 0; i < pares.length; i++) {  
        String pair = pares[i];  
        String[] keyValue = pair.split("=");  
        datos.put(keyValue[0], keyValue[1]);  
    }  
  
    // Se setean a los TextView correspondientes  
    txtNombre.setText(String.valueOf(datos.get("nombre")));  
    txtApellido_s.setText(String.valueOf(datos.get("apellido_s")));  
    txtUsername.setText(String.valueOf(datos.get("nombre_usuario")));  
    txtNacimiento.setText(String.valueOf(datos.get("nacimiento")));  
}
```

app-alumnos

Nombre: David

Apellido(s): BetancourtMontellano

Username: dbetm

Nacimiento: 1998

5) Obtener cambios en tiempo real

- 1) Agregar al diseño de **activity_datos_alumno.xml** lo siguiente:

Apellido(s): TextView

Username: TextView

Nacimiento: TextView

LANZAR AGENTE DE ESCUCHA

Acción escuchada:

6) Obtener cambios en tiempo real

2) Agregar evento onclick al botón, que mande llamar el método,

lanzarAgente(View v)



```
<Button
    android:id="@+id/btnLanzarAgente"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true"
    android:layout_marginBottom="171dp"
    android:text="@string/lanzar_agente"
    android:onClick="lanzarAgente"/>
```

3) Crear el método en la clase **DatosAlumno**, y al inicio deshabilitar el botón para ya no permitir que se vuelva a lanzar otro agente.

```
public void lanzarAgente(View v) {
    // Referencia al botón
    Button btnLanzarAgente = (Button) findViewById(R.id.btnLanzarAgente);
    // Se deshabilita
    btnLanzarAgente.setEnabled(false);
}
```


6) Obtener cambios en tiempo real

4) Agregar el atributo de la BD de Firestore.

```
public class DatosAlumno extends AppCompatActivity {  
    // Atributos  
    FirebaseFirestore db;
```

5) Después agregar otro método **setup()**, tal como se hizo en el **MainActivity**.

```
private void setup() {  
    this.db = FirebaseFirestore.getInstance();  
}
```

6) Obtener cambios en tiempo real

6) Llamar el método **setup()** desde **lanzarAgente(View v)**.

```
public void lanzarAgente(View v) {  
    // Referencia al botón  
    Button btnLanzarAgente = (Button) findViewById(R.id.btnLanzarAgente);  
    // Se deshabilita  
    btnLanzarAgente.setEnabled(false);  
  
    // Para inicializar "bd" en esta actividad.  
    setup();  
}
```

7) Crear la referencia de la colección a escuchar:

```
// Se obtiene el objeto referencia de la colección de interés  
CollectionReference collRef = db.collection(s: "alumnos");
```

6) Obtener cambios en tiempo real

8) Llamar el método addSnapshotListener(...)

```
// Se manda llamar el método que agrega el agente de escucha en tiempo real
collRef.addSnapshotListener(new EventListener<QuerySnapshot>() {
    @Override
    public void onEvent(QuerySnapshot snapshots, FirebaseFirestoreException e) {
        if (e != null) { // Si error es diferente de nulo, el agente da error
            Toast.makeText(getApplicationContext(), text: "Error: " + e,
                Toast.LENGTH_LONG).show();
            return;
        }
        // Se recorren las instancias que presentan cambios
        for(DocumentChange dc : snapshots.getDocumentChanges()) {
            TextView txtAccion = findViewById(R.id.txtAccion);
            switch (dc.getType()) {
                case ADDED: // un alumno se agregó
                    txtAccion.setText(String.valueOf(
                        dc.getDocument().getData().get("nombre") +
                        " se ha agregado."
                    ));
                    Log.d( tag: "Agrega", msg: "New city: " + dc.getDocument().getData());
                    break;
                case MODIFIED: // un alumno se modificó en alguno de sus campos
                    txtAccion.setText(String.valueOf(
                        dc.getDocument().getData() +
                        " se ha modificado."
                    ));
                    Log.d( tag: "Modifica", msg: "Modified city: " + dc.getDocument().getData());
                    break;
                case REMOVED: // un alumno se eliminó
                    txtAccion.setText(String.valueOf(
                        dc.getDocument().getData() +
                        " se ha eliminado."
                    ));
                    Log.d( tag: "Elimina", msg: "Removed city: " + dc.getDocument().getData());
                    break;
            }
        }
    }
});
```

6) Obtener cambios en tiempo real

9) Lanzar la app y hacer pruebas (borrar, actualizar y agregar) desde la consola de Firebase.

The screenshot shows the 'Agregar un documento' (Add document) interface in the Firebase console. The path is set to 'Ruta superior /alumnos'. The document ID is set to 'ID automático'. Two fields are added: 'nombre' with value 'Daniela' and 'apellido_s' with value 'Chávez'. The interface includes a 'Cancelar' (Cancel) button and a 'Guardar' (Save) button.

Agrega un documento

Ruta superior
/alumnos

ID de documento ?

ID automático

Campo	Tipo	Valor	
nombre	string	Daniela	-
apellido_s	string	Chávez	-

+ Agregar campo

Cancelar Guardar

