

Mobile Application Development

Produced
by

David Drohan (ddrohan@wit.ie)

Department of Computing & Mathematics
Waterford Institute of Technology

<http://www.wit.ie>



Waterford Institute of Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE





Android & Firebase

Part 4

Firebase Integration





Agenda

- ❑ Firebase history
- ❑ The all new Firebase
- ❑ Real-time database
- ❑ Authentication
- ❑ Storage
- ❑ Remote config
- ❑ Hosting
- ❑ Crash reporting
- ❑ Test lab
- ❑ Firebase cloud messaging
- ❑ Dynamic links
- ❑ App indexing
- ❑ Analytics
- ❑ CoffeeMate Highlights & Demos along the way...



Agenda

- ❑ Firebase history
- ❑ The all new Firebase
- ❑ **Real-time database**
- ❑ Authentication
- ❑ Storage
- ❑ Remote config
- ❑ Hosting
- ❑ Crash reporting
- ❑ Test lab
- ❑ Firebase cloud messaging
- ❑ Dynamic links
- ❑ App indexing
- ❑ Analytics
- ❑ CoffeeMate Highlights & Demos along the way...



Database



Goodbye RDBMS...



Real Time Database

- ❑ Unlike RDBMS, data is stored as JSON. It is no-SQL JSON database.
- ❑ What makes it real time is its ability to notify listeners of any change in data.
- ❑ Whenever any change is made in the JSON database structure, the firebase SDK notifies the app.
- ❑ So you can forget about REST API calls, connectivity checks, 3rd party libraries and polling.



The Core Magic Of Firebase *





Saving Data In Firebase

❑ Get database reference (your base node)

- `DatabaseReference mDBRef = FirebaseDatabase.getInstance().getReference();`

❑ Point it to the right JSON node

- `mDBRef = mDBRef.child("mydb").child("table1");`

Create a Key

❑ Set the value at the pointed node

- `mDBRef.child('row1').setValue(myPOJO);`

Firebase takes
care of the Key

❑ Or push the value to create a unique key and set the value

- `mDBRef.push().setValue(myPOJO);`



View Your Saved Data

- ❑ Log on to <https://firebase.google.com>.
- ❑ Go to console and select your project.
- ❑ Hit database and select data tab

The screenshot shows the Firebase console interface. On the left, a sidebar lists various services: Authentication, Database (which is selected and highlighted with a red box), Storage, Hosting, Functions, Test Lab, Crash Reporting, and Performance. Below this is a 'GROW' section with Notifications and Remote Config. The main area shows a database tree under the project 'coffeematefbi'. The 'user-coffees' node contains a child node '1SQVbMgN5bcLg9JXgG0ts3DNOAV2', which in turn has a child node '-Kpt-S0GNxNFpsOBSzS-' containing the following data:

- address: "Co. Waterford X91 W29R Ireland"
- favourite: false
- googlephoto: "https://lh5.googleusercontent.com/-AXr-7Z4gX7k/..."
- latitude: 52.24
- longitude: -7.16



Retrieving Data

- ❑ Data is retrieved via callbacks listeners.
- ❑ There are mainly 2 types of listeners
 - [ValueEventListener](#) – get entire child structure.
 - [ChildEventListener](#) – get only the child that got changed / added or deleted.
- ❑ Attach these listeners to [Database reference](#) we saw earlier.
- ❑ Both of these listeners are called once, so app can get the data and prepare UI



Attaching Data Listeners

- ❑ ValueEventListeners can be added in 2 ways:
 - **addListenerForSingleValueEvent (valEvListener);** //next slide
 - ◆ *Get the entire data snapshot only once*
 - **addValueEventListener(valEvListener);**
 - ◆ *Get entire data snapshot whenever there is a change in any of the child nodes*
- ❑ ChildEventListener can be added in only one way:
 - **addChildEventListener(childEvListener)**
 - ◆ *Get updates as child nodes*



Retrieving Data As POJO

```
public ChildEventListener childEvListener = new ChildEventListener() {  
  
    public void onChildAdded(DataSnapshot dataSnapshot, String s) {  
        MyPOJO m = dataSnapshot.getValue(MyPOJO.class);  
    }  
  
    public void onChildChanged(DataSnapshot dataSnapshot, String s) {}  
  
    public void onChildRemoved(DataSnapshot dataSnapshot) {}  
  
    public void onChildMoved(DataSnapshot dataSnapshot, String s) {}  
  
    public void onCancelled(DatabaseError databaseError) {}  
};
```



Retrieving Data As POJO

```
public ValueEventListener valEvListener = new ValueEventListener() {  
    @Override  
    public void onDataChange(DataSnapshot dataSnapshot) {  
        MyPOJO m = dataSnapshot.getValue(MyPOJO.class);  
    }  
  
    @Override  
    public void onCancelled(DatabaseError databaseError) {}  
};
```



Filtering Data

- ❑ Set DatabaseReference to the right parent node.

- Db = base.child('Institute').child('year');

- ❑ To get all students with major = 'Computing'.

- Set orderByChild('major').
 - Set equalTo('Computing');
 - Add listeners

```
Db.orderByChild('major').equalTo('Computing')  
.addChildEventListener(childEvListener);
```



Pagination

- ❑ Create a [Query](#) object or keep using [DatabaseReference](#).
Query is super class of DatabaseReference class.
- ❑ Set `orderBy` some child key (and filter if needed)
- ❑ Set limits (`startAt`, `limit` etc.)
- ❑ Attach data listeners
- ❑ And done. Appropriate callback will be called when operation is complete.



Pagination Code

```
// class level
final int limit = 50;
int start = 0;

// event level
Query userListQuery = userDBRef.orderByChild("email")
.limitToFirst(limit).startAt(start);
userListQuery.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot d) {
        // Do something
        start += (limit+1);
```



Firebase Realtime Database



Quick DEMO...



CoffeeMateFBI 1.0

**Setup
&
Code
Highlights**



1. Create a Firebase Project *

The screenshot shows the Firebase console homepage. At the top, there's a navigation bar with various links like Lynda.com, Android Development, and a search bar for 'console.firebaseio.google.com'. Below the bar, the 'Welcome back to Firebase' message is displayed, along with a small illustration of a person working on a large smartphone. A red box highlights the 'CREATE NEW PROJECT' button, which is blue with white text. To the right of it is a 'IMPORT GOOGLE PROJECT' button. On the left, there's a list of existing projects: 'PaceMaker' with the URL 'pacemaker-a88f1.firebaseio.com'. The overall interface is clean and modern.



1. Create a Firebase Project

The screenshot shows the Firebase console interface on a Mac OS X desktop. The title bar includes standard OS X icons and the URL `console.firebaseio.google.com`. The main navigation bar has items like Lynda.com, Android Development, Electronic Marksheet, etc. A central modal dialog box is open, titled "Create a project". It contains fields for "Project name" (set to "CoffeeMate FBI") and "Country/region" (set to "Ireland"). Below these fields is a note about Firebase Analytics data sharing, with a "Learn more" link. At the bottom of the dialog are "CANCEL" and "CREATE PROJECT" buttons. In the background, the main Firebase dashboard shows a project named "PaceMaker" with the URL `pacemaker-a88f1.firebaseio.com`.



1. Create a Firebase Project *

The screenshot shows the Firebase console homepage. At the top, there's a navigation bar with various links like Lynda.com, Android Development, and a search bar for 'console.firebaseio.google.com'. Below the header, a blue banner says 'Welcome back to Firebase' with a small illustration of a person working on a large smartphone. There are links for Documentation, Sample code, API reference, and Support. In the center, there's a section titled 'Your projects using Firebase' with two cards. The first card, 'CoffeeMate FBI', is highlighted with a red rounded rectangle. It contains the project name, a preview thumbnail, and the URL 'coffeemate-fbi.firebaseio.com'. The second card, 'PaceMaker', also has a preview thumbnail and the URL 'pacemaker-a88f1.firebaseio.com'. At the bottom right of this section are two buttons: 'CREATE NEW PROJECT' and 'IMPORT GOOGLE PROJECT'.



1. Create a Firebase Project - Overview

The screenshot shows the Firebase console's Overview page for the "CoffeeMate FBI" project. The left sidebar lists various services: Analytics, Authentication, Database, Storage, Hosting, Test Lab, and Crash Reporting under the "DEVELOP" section; and Notifications and Remote Config under the "GROW" section. At the bottom, there's a "Spark Free \$0/month" plan and an "UPGRADE" button. The main content area features a welcome message "Welcome to Firebase! Get started here." and three prominent buttons: "Add Firebase to your iOS app" (with an iOS icon), "Add Firebase to your Android app" (with an Android icon), and "Add Firebase to your web app" (with a code icon). Below these are two smaller sections: "Discover Firebase" with an illustration of a person working on a laptop with a chart, and another section showing mobile devices and badges.



1. Create a Firebase Project - Overview

The screenshot shows the Firebase console's Overview page for the project "CoffeeMate FBI". The left sidebar lists various services: Analytics, Authentication, Database, Storage, Hosting, Test Lab, Crash Reporting, Notifications, and Remote Config. A "Spark" section indicates a free \$0/month usage. The main area displays four service cards: Analytics (with an illustration of a person at a laptop), Authentication (with an illustration of user profiles), Database (with an illustration of servers), and Storage (with an illustration of files and documents). Each card has a "Learn more" and "GET STARTED" button.

console.firebaseio.google.com/project/coffeemate-fbi/overview

Firebase CoffeeMate FBI Overview Go to docs

Overview

Analytics

Get detailed analytics to measure and analyse how users engage with your app

[Learn more](#) [GET STARTED](#)

Authentication

Authenticate and manage users from a variety of providers without server-side code

[Learn more](#) [GET STARTED](#)

Database

Store and sync data in real time across all connected clients

[Learn more](#) [GET STARTED](#)

Storage

Store and retrieve user-generated content like images, audio and video without server-side code

[Learn more](#) [GET STARTED](#)

Spark Free \$0/month UPGRADE



2. Add Firebase to your Android App *

Welcome to Firebase! Get started here.

- iOS
- Add Firebase to your iOS app
- Android
- Add Firebase to your Android app
- </>
- Add Firebase to your web app

Discover Firebase

Spark
Free \$0/month [UPGRADE](#)



2. Add Firebase to your Android App – Web Key *

The screenshot shows the Firebase Console interface. On the left, a sidebar lists various services: Analytics, Authentication, Database, Storage, Hosting, Test Lab, Crash Reporting, Notifications, Remote Config, Dynamic Links, EARN, and AdMob. The main area is titled 'Settings' and has tabs for GENERAL, CLOUD MESSAGING, ANALYTICS, ACCOUNT LINKING, and SERVICE ACCOUNTS. Under the GENERAL tab, there's a section for 'Your project' with fields for Project name (CoffeeMate FBI), Public-facing name (CoffeeMate FBI), Project ID (coffeemate-fbi), and Web API Key (AlzaSy...). Below this, under 'Your apps', it says 'There are currently no apps in the project. CoffeeMate FBI'. At the bottom, there are three buttons: 'Add Firebase to your iOS app' (with an iOS icon), 'Add Firebase to your Android app' (with an Android icon), and 'Add Firebase to your web app' (with a code icon). The URL in the browser bar is console.firebaseio.google.com/project/coffeemate-fbi/settings/general/.



2. Add Firebase to your Android App *

Add Firebase to your Android app

1 2 3

Register app Download config file Add Firebase SDK

A An OAuth2 client already exists for this package name and SHA-1 in another project. You can omit the SHA-1 for now and [read more about this situation and how to resolve it.](#)

Android package name ?

 (Optional)

App nickname (optional) ?

Debug signing certificate SHA-1 (optional) ?

 (Optional)

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

CANCEL REGISTER APP

in project **CoffeeMateFBI**

Remember to update/add to your google maps key if you change your package name (or else your map won't work!)



2. Add Firebase to your Android App *

Add Firebase to your Android app

1 Register app 2 Download config file 3 Add Firebase SDK

Android Studio instructions Alternatives: [Unity](#) [C++](#)

[Download google-services.json](#)

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

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

google-services.json

Already added the dependencies?
[Skip to the console](#)

CONTINUE



2. Add Firebase to your Android App *

The screenshot shows the Firebase Console interface with the URL `console.firebaseio.google.com/project/coffeemate-fbi-e815e/settings/general/`. The main navigation bar includes links like Lynda.com, Android Development, Electronic Marksheets, lynda.com, solution.md, Triathlon Ireland, Apple Developer, TeamViewer, MapMyRun, Garmin Connect, WIT Catalogue, OneDrive, Google Drive, and a general Firebase Console link.

The left sidebar lists various Firebase services: Analytics, Authentication, Database, Storage, Hosting, Test Lab, Crash Reporting, Notifications, Remote Config, Dynamic Links, AdMob, and EARN.

The central area displays a modal window titled "Add Firebase to your Android app" with three steps: "Enter app details", "Copy config file", and "Add to build.gradle". The third step is currently active.

The instructions in the modal state: "The Google services plugin for [Gradle](#) loads the google-services.json file that you just downloaded. Modify your build.gradle files to use the plugin."

Step 1: "Project-level build.gradle (<project>/build.gradle):"

```
buildscript {  
    dependencies {  
        // Add this line  
        classpath 'com.google.gms:google-services:3.0.0'  
    }  
}
```

Step 2: "App-level build.gradle (<project>/<app-module>/build.gradle):"

```
...  
// Add to the bottom of the file  
apply plugin: 'com.google.gms.google-services'  
include Firebase Analytics by default
```

Step 3: "Finally, press "Sync now" in the bar that appears in the IDE:"

A progress bar at the bottom indicates "Gradle files have changed since last sync" and features a blue "Sync now" button.

At the bottom right of the modal is a blue "FINISH" button.



CoffeeMateFBI.1.0 Dependencies (July/2017) *

```
dependencies {  
    compile fileTree(include: ['*.jar'], dir: 'libs')  
    compile project(':volley')  
    compile 'com.android.support:appcompat-v7:25.4.0'  
    compile 'com.android.support:support-v4:25.4.0'  
    compile 'com.android.support:design:25.4.0'  
    compile 'com.makeramen:roundedimageview:2.2.1'  
    compile 'com.android.support.constraint:constraint-layout:1.0.2'  
    compile 'com.google.code.gson:gson:2.7'  
    compile 'com.google.android.gms:play-services-auth:11.0.2'  
    compile 'com.google.android.gms:play-services-maps:11.0.2'  
    compile 'com.google.android.gms:play-services-location:11.0.2'  
  
    compile 'com.google.firebaseio:firebase-core:11.0.2'  
    compile 'com.google.firebaseio:firebase-auth:11.0.2'  
    compile 'com.google.firebaseio:firebase-database:11.0.2'  
    compile 'com.firebaseui:firebase-ui-database:2.1.0'  
    testCompile 'junit:junit:4.12'  
}
```



FBDBManager – our Firebase Database utility class *

```
public class FBDBManager {  
  
    private static final String TAG = "coffeemate";    □ Our Firebase db reference.  
    public DatabaseReference mFirebaseDatabase;  
    public static String mFBUserId;  
    public FBDBListener mFBDBListener;  
  
    public void open() {  
        //Set up local caching  
        FirebaseDatabase.getInstance().setPersistenceEnabled(true);  
  
        //Bind to remote Firebase Database  
        mFirebaseDatabase = FirebaseDatabase.getInstance().getReference();  
        Log.v(TAG, "Database Connected :" + mFirebaseDatabase);  
    }  
}
```

□ Binding to our db instance.



FBDBManager – usage *

```
public class CoffeeMateApp extends Application
{
    private RequestQueue mRequestQueue;
    private static CoffeeMateApp mInstance;
    //...

    public FBDBManager mFBDBManager;

    //...
    @Override
    public void onCreate() {
        super.onCreate();
        Log.v("coffeemate", "CoffeeMate App Started");
        mInstance = this;
        mRequestQueue = Volley.newRequestQueue(getApplicationContext());
        Log.v("coffeemate", "Adding Firebase offline persistence...");
        mFBDBManager = new FBDBManager();
        mFBDBManager.open();
    }
}
```

- ❑ Our utility class reference (inside our Application object class).
- ❑ Creating & ‘Opening’ a connection to our Firebase db.



FBDBManager – our Firebase Database utility class *

- ❑ ‘Query’ing the ‘*mFBUserId*’ node of the ‘*user-coffees*’ node.

```
public Query getAllCoffees()
{
    Query query = mFirebaseDatabase.child("user-coffees").child(mFBUserId)
        .orderByChild("rating");

    return query;
}

public Query getFavouriteCoffees()
{
    Query query = mFirebaseDatabase.child("user-coffees").child(mFBUserId)
        .orderByChild("favourite").equalTo(true);

    return query;
}
```

- ❑ As above, but only where the ‘*favourite*’ field is ‘*true*’.



FBDBManager – usage (in CoffeeFragment) *

```
@Override  
public void onResume() {  
    super.onResume();  
  
    query = app.mFBDBManager.getAllCoffees();  
  
    if(favourites)  
        query = app.mFBDBManager.getFavouriteCoffees();  
  
    updateUI(query);  
}
```

- Returning a ‘query’ of all coffees, (or favourite coffees) which we pass to our custom firebaseUI adapter via *updateUI()*



FBDBManager – our Firebase Database utility class *

- Adding an ‘ValueEventListener’ and triggering our callback

```
public void getACoffee(final String coffeeKey)
{
    mFirebaseDatabase.child("user-coffees").child(mFBUserId).child(coffeeKey)
        .addValueEventListener(
            new ValueEventListener() {
                @Override
                public void onDataChange(DataSnapshot dataSnapshot) {
                    Log.v(TAG, "The read Succeeded: " + dataSnapshot.toString());
                    mFBDBListener.onSuccess(dataSnapshot);
                }
                @Override
                public void onCancelled(DatabaseError firebaseError) {
                    Log.v(TAG, "The read failed: " + firebaseError.getMessage());
                    mFBDBListener.onFailure();
                }
            });
}
```



FBDBManager – usage (in EditFragment) *

```
@Override  
public void onResume() {  
    super.onResume();  
  
    app.mFBDBManager.attachListener(this);  
  
    if(getArguments() != null) {  
        coffeeKey = getArguments().getString("coffeeKey");  
        app.mFBDBManager.getACoffee(coffeeKey);  
    }  
    titleBar = (TextView) getActivity()  
        .findViewById(R.id.recentAddedBarTextView);  
    titleBar.setText("Update a Coffee");  
}
```

- ☐ Retrieving the ‘coffeeKey’ from the bundle and calling ‘getACoffee()’ - which in turn triggers the callback on the edit screen



Firebase Console (actual data)

The image shows the Firebase Realtime Database console on the left and an Android emulator on the right.

Firebase Realtime Database Screenshot:

- The database structure under "user-coffees" includes nodes for multiple coffee shops, such as "1SQVbMgN5bcLg9JXgG0ts3DNOAV2" and "C3PpngqTi8OxuHvNtG4wZsVMhZv2".
- The node "1SQVbMgN5bcLg9JXgG0ts3DNOAV2" is highlighted with a red box and contains the following data:
 - KpRYrfoaJaqJZGt9Mx6
 - address: "191 Hennessy's Road Waterford X91 PXA4"
 - favourite: false
 - googlephoto: "https://lh5.googleusercontent.com/-AXr-7Z4gX7k/..."
 - latitude: 52.25
 - longitude: -7.126
 - name: "Firebase Coffee"
 - price: 1.99
 - rating: 2
 - shop: "Fire Station"
 - uid: "1SQVbMgN5bcLg9JXgG0ts3DNOAV2"
 - usertoken: "113437677814759908125"
 - KpRZ845g8sLB4TDNo0w
 - address: "5 Lismore Park Waterford Ireland"
 - favourite: false
 - googlephoto: "https://lh5.googleusercontent.com/-AXr-7Z4gX7k/..."
 - latitude: 52.25
 - longitude: -7.1399983
 - name: "Coffee Latte"
 - price: 2.49
 - rating: 2
 - shop: "Aldi"
 - uid: "1SQVbMgN5bcLg9JXgG0ts3DNOAV2"
 - usertoken: "113437677814759908125"

Android Emulator Screenshot:

- The emulator displays the "Recently Added Coffee's" screen from the "CoffeeMateFBI" app.
- The list shows two items:
 - 1. Firebase Coffee (Fire Station) - €1.99 (2.99+)
 - 2. Coffee Latte (Aldi) - €2.49 (2.99+)
- The first item is highlighted with a red box.



Firebase Console (actual data)

The screenshot shows the Firebase Realtime Database interface and an Android emulator side-by-side.

Firebase Realtime Database Screenshot:

- The database structure under `user-coffees` includes entries for users with IDs `1SQVbMgN5bcLg9JXgG0ts3DNOAV2`, `-KpRYrfoAJaqJZGt9Mx6`, `-KpRZ845g8sLB4TDN0oW`, `C3PpngqTl8OxuHvNtG4wZsVMhZv2`, and `wotuVu5IZjh0HRcCHRkJgsWh9y1`.
- The entry for user `-KpRYrfoAJaqJZGt9Mx6` contains a child node `favourite: true`, which is highlighted with a red box.
- The entry for user `-KpRZ845g8sLB4TDN0oW` contains a child node `favourite: false`.
- Both entries also contain other fields like address, latitude, longitude, name, price, rating, shop, uid, and usertoken.

Android Emulator Screenshot:

- The emulator displays the `CoffeeMateFBI` application.
- The app shows a list of "Recently Added Coffee's" with two items:
 - Firebase Coffee** (marked with a star) - **Fire Station** - **€1.99** - **2.0+** - **X**
 - Coffee Latte** (marked with a star) - **Aldi** - **€2.49** - **2.0+** - **X**
- The item **Firebase Coffee** is also highlighted with a red box.



Pricing

<https://firebase.google.com/pricing/>

Products	Spark Plan Generous limits for hobbyists Free	Flame Plan Fixed pricing for growing apps \$25/month	Blaze Plan Calculate pricing for apps at scale Pay as you go
Free Products Authentication (except Phone Auth), Analytics, App Indexing, Dynamic Links, Invites, Remote Config, Cloud Messaging (FCM), Performance Monitoring, and Crash Reporting.	Included	Included Free	Included Free
Realtime Database Simultaneous connections	100	100K/instance	100K/instance
GB stored	1 GB	2.5 GB	\$5/GB
GB downloaded	10 GB/month	20 GB/month	\$1/GB
Automated backups			



Thanks!



A black and white cartoon illustration of a hand with three fingers pointing towards a smiling sun-like character. The sun has a simple face with two dots for eyes and a curved line for a mouth. It is surrounded by small clouds. The hand is positioned as if presenting or highlighting the sun.



Some important points though...

- ❑ Do not think RDBMS, think JSON. How data should be structured is very important.
- ❑ Firebase has a recycler view, that integrates with real time database smoothly without any listeners. (FirebaseUI)
- ❑ Test lab which is available in paid plan (Blaze), is an amazing feature for testing your app on different real and virtual devices (next section)
- ❑ Set developer mode to true when testing Remote Config (next section).



References & Links

- ❑ [Presentation by Kaushal Dhruw & Shakti Moyal 2016](#)
- ❑ <https://firebase.google.com>
- ❑ Demo app available at <https://goo.gl/WBP5fR>



Questions?