



Firestore



Firestore

Firebase คือ..

Firebase เป็น Project ถูกออกแบบมาให้เป็น API และ CloudStorageสำหรับพัฒนา Realtime Application รองรับหลาย Platform เบื้องต้นล่าสุดก็มีให้ใช้พัฒนากับ 3 Platform คือ IOS App, Android App, Web App

Firebase มีบริการอะไรให้บ้าง..

1 Firebase Analytics

บริการวิเคราะห์ข้อมูล ดึงเทคโนโลยีมาจาก Google Analytics แคมเปญเปิดให้ใช้ฟรีแบบไม่จำกัดปริมาณข้อมูลใดๆ

2 Firebase Cloud Messaging (FCM)

ระบบส่งข้อความแจ้งเตือน ใช้งานฟรีไม่จำกัดปริมาณข้อความ

3 Firebase Storage

บริการพื้นที่เก็บข้อมูล เอาไว้เก็บภาพ วิดีโอ หรือไฟล์ขนาดใหญ่จากแอปของผู้ใช้ สร้างอยู่บน Google Cloud Storage

4 Firebase Remote Config

ตัวช่วยอัปเดตคอนฟิกของแอป สำหรับปรับแต่งค่าต่างๆ ในแอปจากระยะไกล (เช่น เกมที่อยากปรับสมดุลของเกมตลอดเวลา) สามารถใช้ร่วมกับ Firebase Analytics เพื่อกำหนดผู้ใช้งานแยกเป็นกลุ่มๆ ได้

Firebase มีบริการอะไรให้บ้าง..

5 Firebase Crash Reporting

ตัวรายงานการแครชของแอป รองรับทั้ง iOS และ Android

6 Firebase Test Lab for Android

บริการทดสอบแอปบนฮาร์ดแวร์จริง

7 Firebase Notifications

Firebase Notifications เป็นคอนโซลสำหรับนักพัฒนา เพื่อยิงข้อความผ่าน FCM ไปยังผู้ใช้ สำหรับโปรโมทหรือกระตุ้นให้ผู้ใช้กลับมาเปิดแอปของเรา (เช่น แจกของในเกม)

8 Firebase Dynamic Links

Firebase Dynamic Links บริการ URL กลางที่สามารถชี้ทางไปยังเพจต่างๆ แปรผันตามอุปกรณ์หรือคุณสมบัติของผู้ใช้ (เช่น แต่ละประเทศกดลิงก์เดียวกัน เข้าคนละเพจกัน)

Firebase มีบริการอะไรให้บ้าง..

9

Firebase Invites

ระบบเชิญเพื่อนมาใช้แอป มีฟีเจอร์
referral คนชวนได้สิทธิประโยชน์

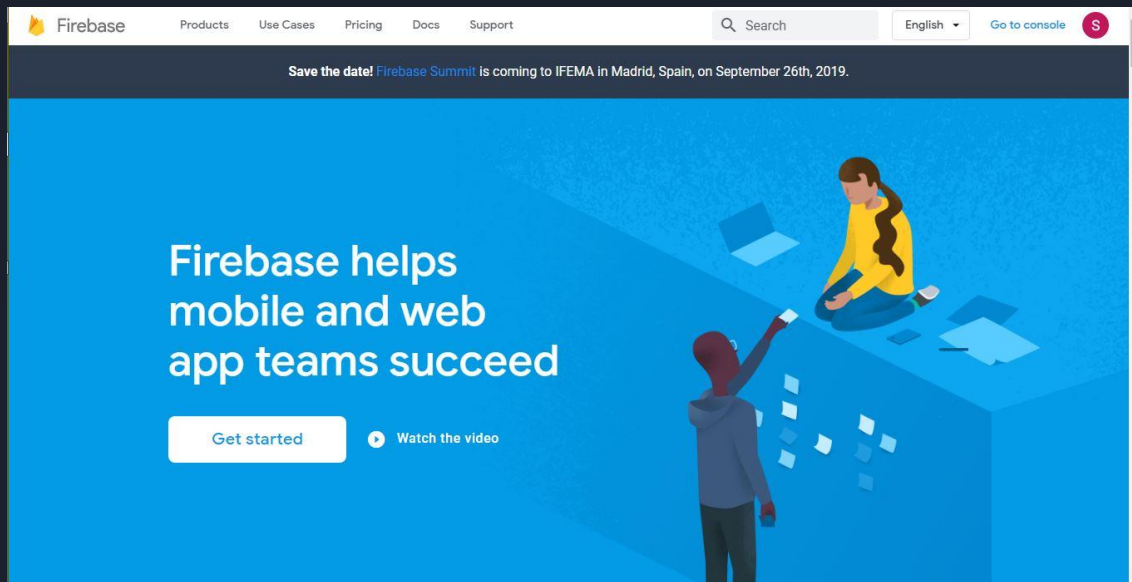
1
0

Firebase App Indexing

เปลี่ยนชื่อมาจาก Google App
Indexing ที่ช่วยให้ Google
Search ค้นเจอเนื้อหาภายในแอป

ขั้นตอนการสมัครใช้งานFirebase

ขั้นตอนแรก เข้าไปที่หน้าเว็บไซต์ <https://firebase.google.com/>



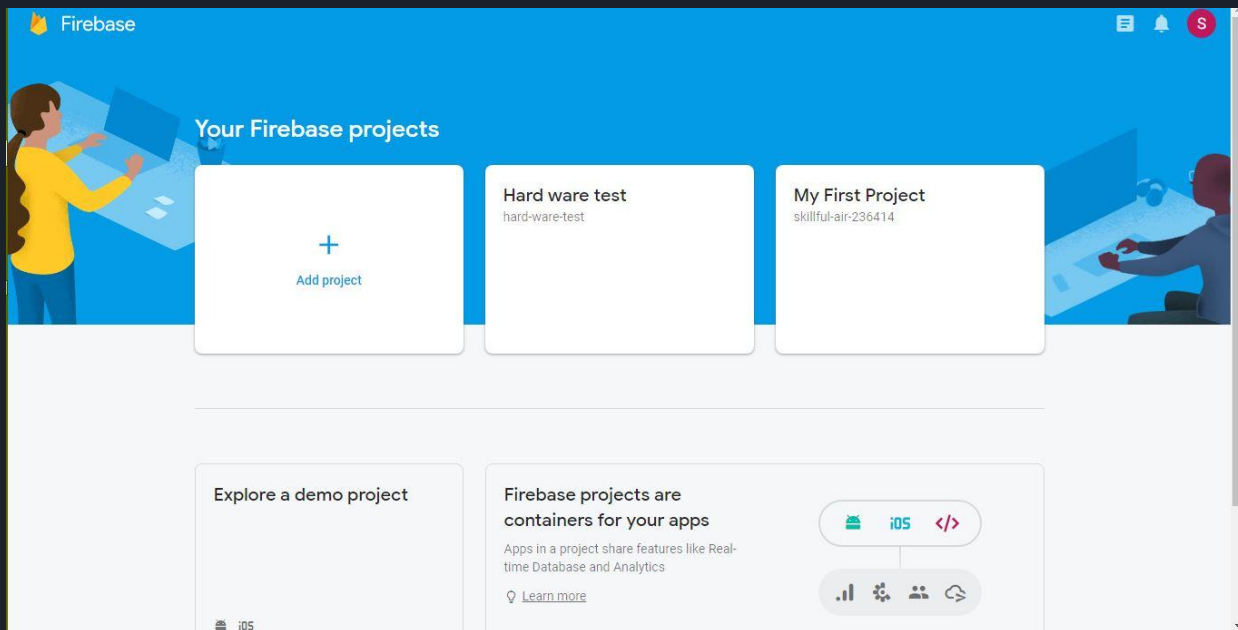
Sign in ให้เรียบร้อย แล้วเข้าไปที่ Go to console

English

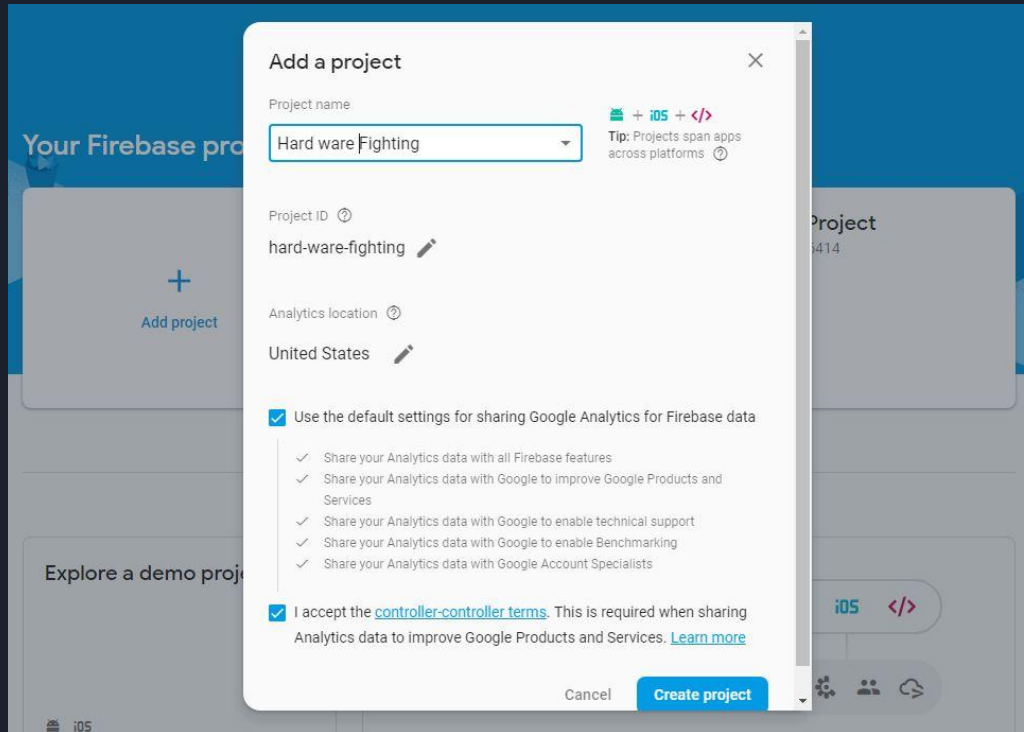
Go to console

S

ต่อไปจะเป็นการสร้าง Project กัน พร้อมรียังไปกันเลย.....



คลิกเลือกที่ปุ่ม Add PROJECT จะมีหน้าจอเด้งมาให้กรอก Project Name และเลือก Country/Region เราก็เลือกตามใจ จากนั้นกดปุ่ม CREATE PROJECT



The screenshot shows the 'Add a project' dialog box in the Firebase console. The dialog has a title bar with a close button (X). It contains the following fields and options:

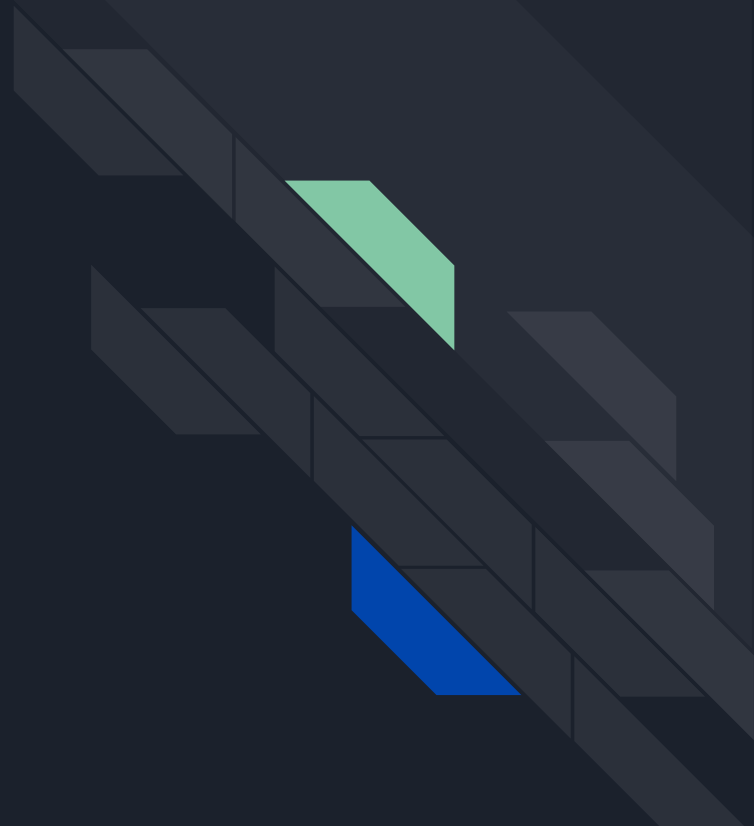
- Project name:** A text input field with the value 'Hard ware Fighting' and a dropdown arrow. To the right, there are icons for Android, iOS, and a code icon, along with a tip: 'Tip: Projects span apps across platforms'.
- Project ID:** A text input field with the value 'hard-ware-fighting' and an edit icon.
- Analytics location:** A text input field with the value 'United States' and an edit icon.
- Default settings for sharing Google Analytics for Firebase data:** A section with a checked checkbox and a list of five items, each with a checked checkbox:
 - 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
- Acceptance:** A checked checkbox followed by the text: 'I accept the [controller-controller terms](#). This is required when sharing Analytics data to improve Google Products and Services. [Learn more](#)'.

At the bottom of the dialog, there are two buttons: 'Cancel' and 'Create project'.

หลังจาก Add PROJECT แล้ว ก็จะได้หน้า console ของ Project
ที่เราสร้างไปเ้ย!!!!!!!!!!

The screenshot shows the Firebase console interface. On the left is a dark sidebar with the 'Firebase' logo at the top. Below the logo are sections: 'Project Overview' (with a home icon and settings gear), 'Develop' (containing links for Authentication, Database, Storage, Hosting, Functions, and ML Kit), 'Quality' (containing links for Crashlytics, Performance, and Test Lab), and 'Analytics' (containing a 'Spark' section showing 'Free \$0/month' and an 'Upgrade' button). The main content area has a blue header with the project name 'Hard ware Fighting' and a 'Spark plan' button. The main heading is 'Get started by adding Firebase to your app', followed by a subtext: 'Our core SDK unlocks most Firebase features and includes Analytics for iOS and Android apps'. Below this are icons for iOS, Android, and code, with a button 'Add an app to get started'. An illustration of two people, one pointing at a screen and the other holding a phone, is on the right. At the bottom, a white banner reads 'Store and sync app data in milliseconds' with a close button 'X'.

การเชื่อมต่อ Arduino ESpino 32 เข้ากับ Firebase



การเชื่อมต่อ Arduino กับ Firebase เข้าด้วยกัน โดยมี WiFi และการเข้ารหัส เป็น
ตัวกลางในการเชื่อมต่อ ทั้งสองอย่างเข้าด้วยกัน

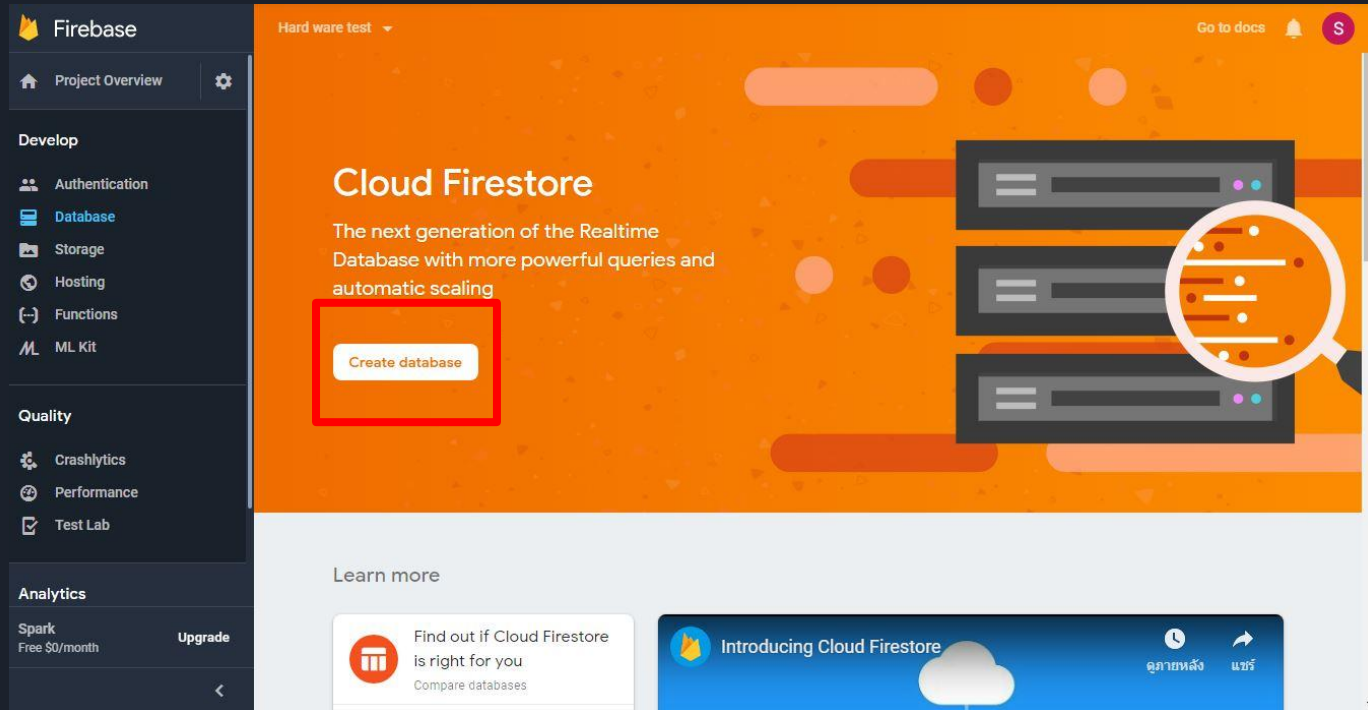


Firebase

Firebase

เราจะมาเริ่มทำส่วนของ Firebase กันก่อน

เข้ามาที่หน้า Console ของเรา เลือกที่ Database จากนั้นในคลิกที่ Create database



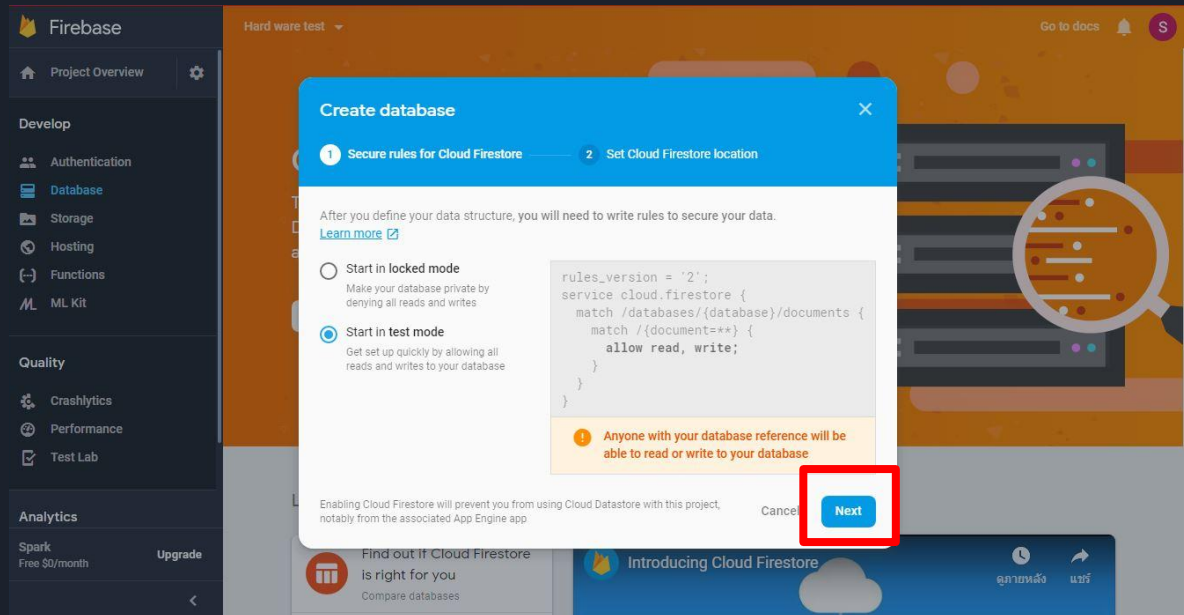
Firebase

เมื่อคลิกที่ Create database แล้วจะมีหน้าจอตั้งขึ้นมา

เข้ามาที่หน้า Console ของเรา เลือกที่ Database จากนั้นในคลิกที่ Create database

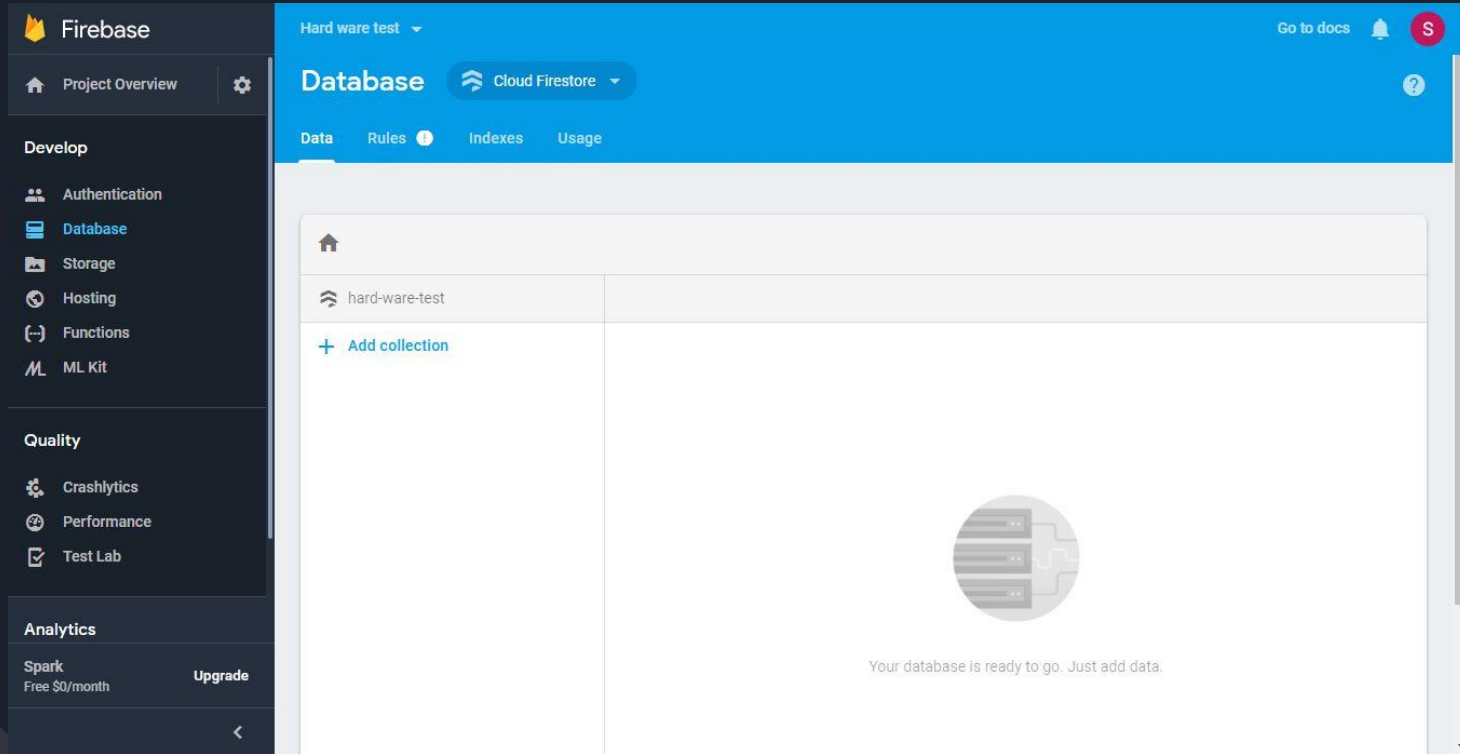
ส่วนที่1 ให้เลือก Start in test mode แล้วกด Next

ส่วนที่2 ให้เลือก Location เป็น asia-east2 จากนั้นให้กด Done



Firebase

เป็นอันเสร็จเรียบร้อย จะได้หน้าตาเป็นประมาณนี้



Firebase

Database ที่มีให้ใช้ โดยมีด้วยกัน 2 แบบ คือ

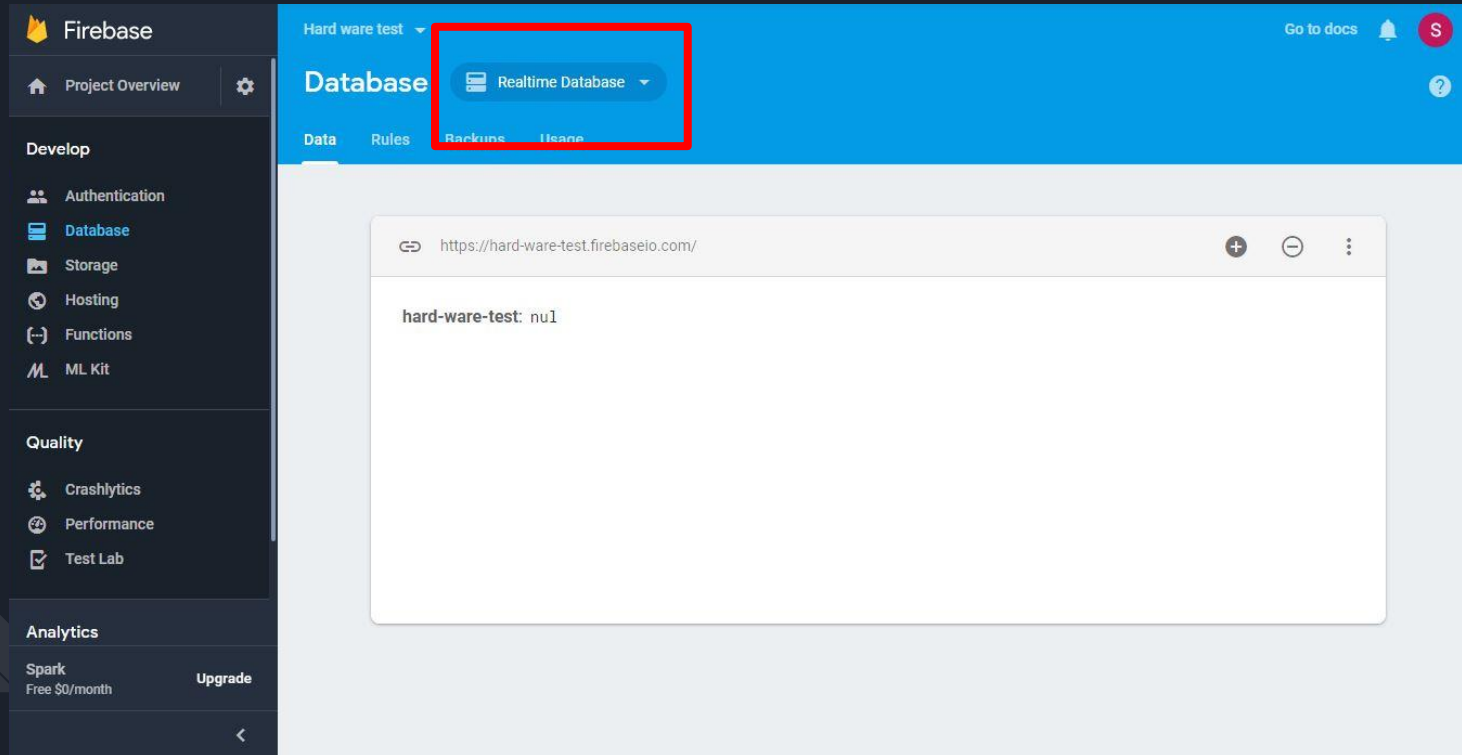
- Cloud Firestore (CFS) คือ บริการฐานข้อมูลแบบ NoSql ที่จัดเก็บในรูปแบบ document
- Realtime database (RDB) คือ บริการฐานข้อมูลแบบ NoSql ที่จัดเก็บในรูปแบบ JSON

ซึ่งเราจะใช้ RDB ในการ SET ค่าเพื่อเชื่อมต่อกับ Arduino ESPino32



Firestore

ให้เลือกตรงกรอบสีแดง เลือกเป็น Realtime database แล้วจะขึ้นรูปหน้าตาแบบนี้



The screenshot shows the Firebase console interface. On the left is a dark sidebar with the 'Firebase' logo and a list of services: Project Overview, Authentication, Database (highlighted in blue), Storage, Hosting, Functions, ML Kit, Crashlytics, Performance, and Test Lab. The main area has a blue header with 'Hard ware test' and a 'Database' section. In the 'Database' section, a dropdown menu is open, showing 'Realtime Database' selected, which is highlighted by a red rectangular box. Below the header, there are tabs for 'Data', 'Rules', 'Backups', and 'Users'. The main content area displays a web browser window with the URL 'https://hard-ware-test.firebaseio.com/' and the text 'hard-ware-test: nul'.

Firestore

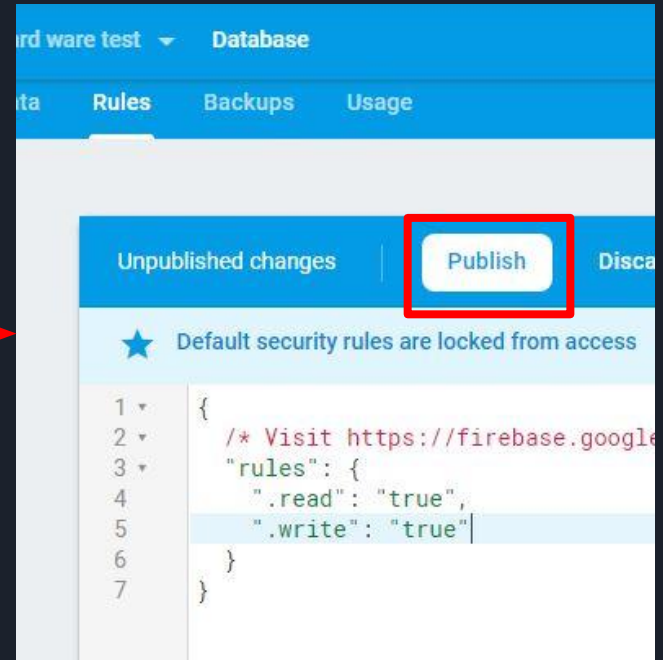
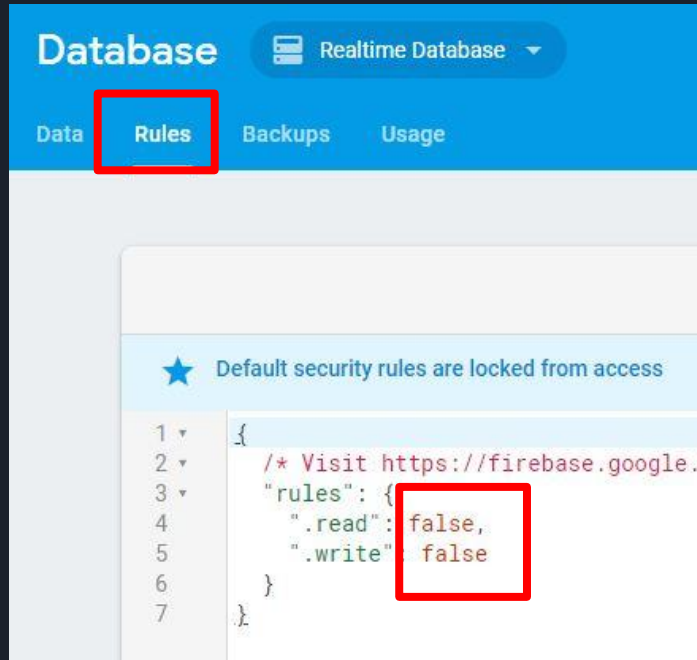
ซึ่ง Realtime database จะเป็นข้อมูลแบบ Tree ซึ่งในที่นี้ยกตัวอย่างเป็น PC เรา โดยข้างในจะมีโฟลเดอร์ย่อยๆ แบ่งออกเป็นหลายๆส่วน ซึ่งในโฟลเดอร์นั้นจะมีข้อมูลที่ แตกต่างออกไป

The screenshot displays the Firebase Realtime Database interface. On the left is a sidebar with navigation options: Project Overview, Develop (Authentication, Database, Storage, Hosting, Functions, ML Kit), Quality (Crashlytics, Performance, Test Lab), and Analytics (Spark, Free \$0/month, Upgrade). The main area shows the 'Database' section for the project 'Hard ware test'. The 'Realtime Database' tab is selected, and the 'Data' sub-tab is active. A tree view is shown for the path 'https://hard-ware-test.firebaseio.com/'. The tree structure is as follows:

- hard-ware-test
 - This PC
 - dive c
 - data: "data"
 - dive d
 - data: "data"
 - google dive: "data"

Firestore

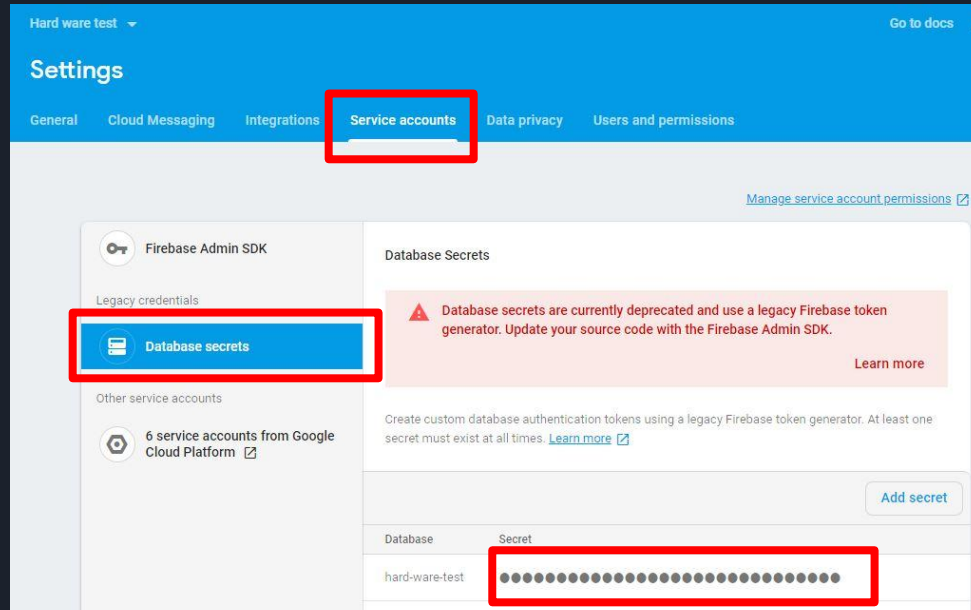
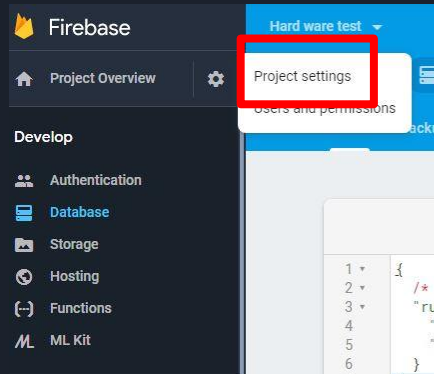
คลิกไปที่ Rules แล้ว แก้ตามนี้นะครับ



Firestore

ต่อไปเราจะมาดู รหัส ที่เราจะไว้ใช้เชื่อมต่อกับ Arduino กัน

เข้าไปที่ Project settings และเลือก ที่ Service accounts เลือกที่ Database secrets เราจะใช้ secret ในการเชื่อมต่อ ซึ่งมันจะปิดไว้ให้ กด Show เพื่อให้ secret มันแสดง






จบการ Set ฝั่ง Firebase

Arduino

ต่อไปจะเป็นการ Set ฝั่งทาง Arduino กันบ้าง มาเริ่มกันเลย..

โดยที่เราจะต้อง Add Library ของ Firebase ก่อน

ตามลิงนี้ <https://github.com/sathitSAPMEK/Library-Firebase> ไป Download หรือ colne ก็ได้
จะได้ไฟล์ .Zip ตามนี้

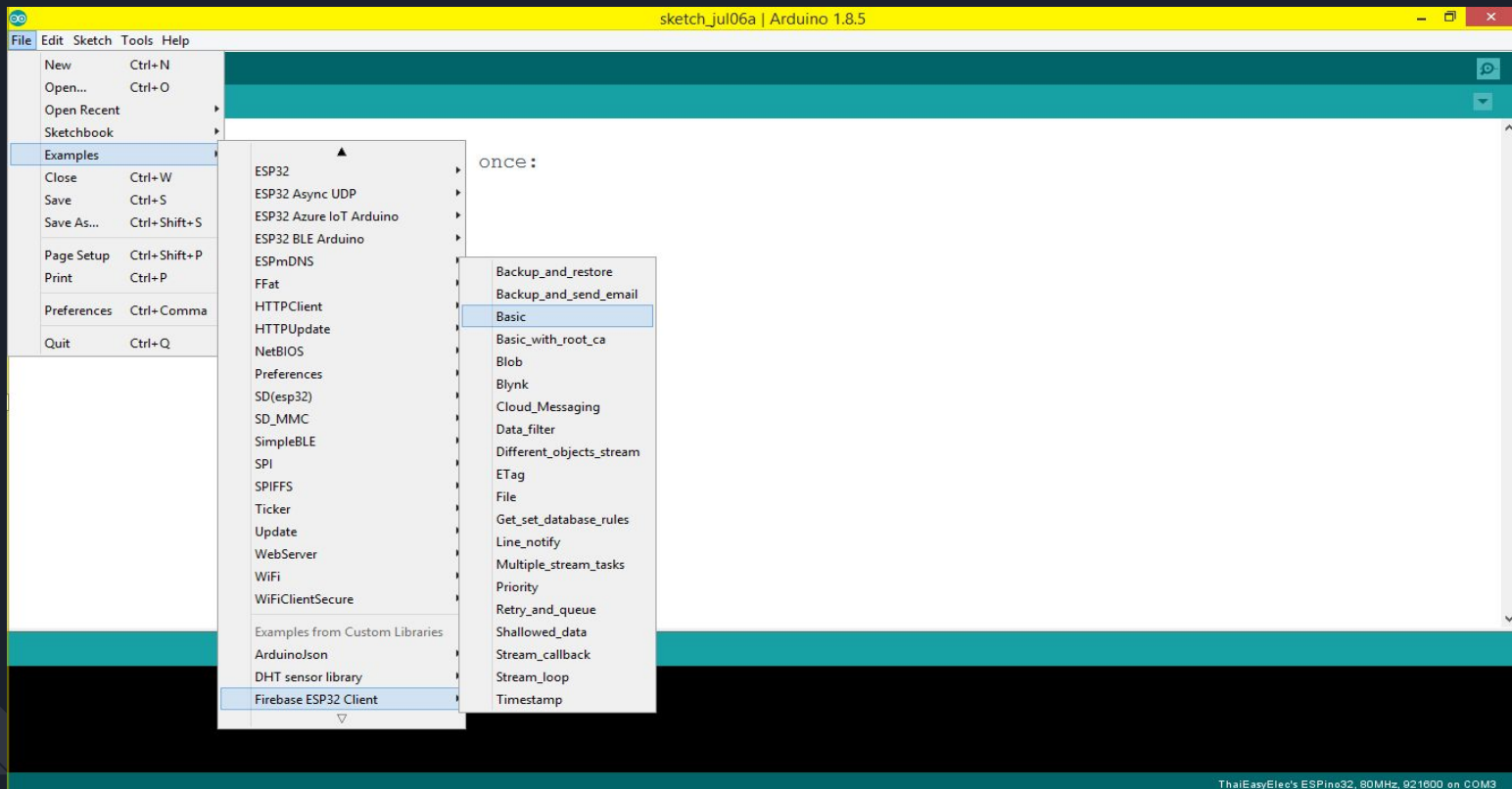
 ArduinoJson-6.11.1.zip	4/7/2562 13:02	ไฟล์เดจอร์ที่บีบอัด	352 KB
 Firebase-ESP32-master.zip	4/7/2562 12:20	ไฟล์เดจอร์ที่บีบอัด	92 KB
 HTTPClientESP32Ex-master.zip	4/7/2562 12:29	ไฟล์เดจอร์ที่บีบอัด	5 KB

ลง Library ทั้ง 3 ให้เรียบร้อย

เมื่อลงเสร็จแล้ว เข้ามาสู่การcode โดยเข้ามาที่ Arduino IDE

Arduino

เข้าไปที่ file > Examples > Firebase ESP32 Client > คลิกที่ Basic



Arduino

ใน Basic จะเป็นการใช้งาน ในการเชื่อมต่อ wifi และ การเชื่อมต่อกับ Firebase
(ซึ่งจะค่อนข้าง งง 555+) ตัวอย่างบางส่วน

```
Basic
#include <WiFi.h>
#include "FirebaseESP32.h"

#define FIREBASE_HOST "YOUR_FIREBASE_PROJECT.firebaseio.com" //Do not include https:// in FIREBASE_HOST
#define FIREBASE_AUTH "YOUR_FIREBASE_DATABASE_SECRET"
#define WIFI_SSID "YOUR_WIFI_AP"
#define WIFI_PASSWORD "YOUR_WIFI_PASSWORD"

//Define Firebase Data object
FirebaseData firebaseData;

void setup()
{
  Serial.begin(115200);
  Serial.println();
  Serial.println();

  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.println("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    delay(1000);
  }
  Serial.println();
  Serial.println("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();
}
```

```
Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
Firebase.reconnectWiFi(true);

//Set database read timeout to 1 minute (max 15 minutes)
Firebase.setReadTimeout(firebaseData, 1000 * 60);
//tiny, small, medium, large and unlimited.
//Size and its write timeout e.g. tiny (1s), small (10s), medium (30s) and large (60s).
Firebase.setWriteSizeLimit(firebaseData, "tiny");

/*
This option allows get and delete functions (PUT and DELETE HTTP requests) works for device connected behind the
Firewall that allows only GET and POST requests.

Firebase.enableClassicRequest(firebaseData, true);
*/

String path = "/ESP32_Test";
String jsonStr;

//Firebase.deleteNode(firebaseData, path);

Serial.println("-----");
Serial.println("Path exist test...");
if (Firebase.pathExist(firebaseData, path))
{
  Serial.println("Path " + path + " exists");
}
else
{
}
```

```
Serial.println("-----");
Serial.println("Set integer test...");

for (uint8_t i = 0; i < 10; i++)
{
  if (Firebase.setInt(firebaseData, path + "/Int/Data" + (i + 1), (i + 1) * 10))
  {
    Serial.println("PASSED");
    Serial.println("PATH: " + firebaseData.dataPath());
    Serial.println("TYPE: " + firebaseData.dataType());
    Serial.println("ETag: " + firebaseData.ETag());
    Serial.println("VALUE: ");
    if (firebaseData.dataType() == "int")
      Serial.println(firebaseData.intData());
    else if (firebaseData.dataType() == "float")
      Serial.println(firebaseData.floatData(), 5);
    else if (firebaseData.dataType() == "double")
      printf("%.9lf\n", firebaseData.doubleData());
    else if (firebaseData.dataType() == "boolean")
      Serial.println(firebaseData.boolData() == 1 ? "true" : "false");
    else if (firebaseData.dataType() == "string")
      Serial.println(firebaseData.stringData());
    else if (firebaseData.dataType() == "json")
      Serial.println(firebaseData.jsonData());
    Serial.println("-----");
  }
  else
  {
    Serial.println("FAILED");
  }
}
```

```
for (uint8_t i = 0; i < 10; i++)
{
  if (Firebase.setDouble(firebaseData, path + "/Double/Data" + (i + 1), ((i + 1) * 10) + 0.123456789))
  {
    Serial.println("PASSED");
    Serial.println("PATH: " + firebaseData.dataPath());
    Serial.println("TYPE: " + firebaseData.dataType());
    Serial.println("ETag: " + firebaseData.ETag());
    Serial.println("VALUE: ");
    if (firebaseData.dataType() == "int")
      Serial.println(firebaseData.intData());
    else if (firebaseData.dataType() == "float")
      Serial.println(firebaseData.floatData(), 5);
    else if (firebaseData.dataType() == "double")
      printf("%.9lf\n", firebaseData.doubleData());
    else if (firebaseData.dataType() == "boolean")
      Serial.println(firebaseData.boolData() == 1 ? "true" : "false");
    else if (firebaseData.dataType() == "string")
      Serial.println(firebaseData.stringData());
    else if (firebaseData.dataType() == "json")
      Serial.println(firebaseData.jsonData());
    Serial.println("-----");
  }
  else
  {
    Serial.println("FAILED");
    Serial.println("REASON: " + firebaseData.errorReason());
    Serial.println("-----");
  }
}
```

```
if (Firebase.getInt(firebaseData, path + "/Double/Data" + (i + 1)))
{
  Serial.println("PASSED");
  Serial.println("PATH: " + firebaseData.dataPath());
  Serial.println("TYPE: " + firebaseData.dataType());
  Serial.println("ETag: " + firebaseData.ETag());
  Serial.println("VALUE: ");
  if (firebaseData.dataType() == "int")
    Serial.println(firebaseData.intData());
  else if (firebaseData.dataType() == "float")
    Serial.println(firebaseData.floatData(), 5);
  else if (firebaseData.dataType() == "double")
    printf("%.9lf\n", firebaseData.doubleData());
  else if (firebaseData.dataType() == "boolean")
    Serial.println(firebaseData.boolData() == 1 ? "true" : "false");
  else if (firebaseData.dataType() == "string")
    Serial.println(firebaseData.stringData());
  else if (firebaseData.dataType() == "json")
    Serial.println(firebaseData.jsonData());
  Serial.println("-----");
}
else
{
  Serial.println("FAILED");
  Serial.println("REASON: " + firebaseData.errorReason());
  Serial.println("-----");
}
```

Arduino

ซึ่งพีได้นำโค้ดจาก Basic นำส่วนที่สำคัญในการเชื่อมต่อกับ Firebase มาไว้แล้วตามรูปที่ต่อจากนี้ไป

ลิงสำหรับ Firebase https://github.com/sathitSAPMEK/Library_Firebase-Functions_Firebase-Code_Firebase

FrieBases §

```
#include <WiFi.h>
#include "FirebaseESP32.h"

#define FIREBASE_HOST "Name Project.firebaseio.com" //Do not include https:// in FIREBASE_HOST
#define FIREBASE_AUTH "Secret"
#define WIFI_SSID "User Name"
#define WIFI_PASSWORD "Password"

FirebaseData firebaseData;

void setup() {
  Serial.begin(115200);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    delay(300);
  }
}
```

Database

Realtime Database

Data Rules Backups Usage

https://hard-ware-test.firebaseio.com/

hard-ware-test

This PC

dive c

data: "date"

dive d

data: "date"

google dive: "date"

Arduino

```
Serial.println();  
Serial.print("Connected with IP: ");  
Serial.println(WiFi.localIP());  
Serial.println();  
  
Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);  
Firebase.reconnectWiFi(true);  
Firebase.setReadTimeout(firebaseData, 1000 * 60);  
Firebase.setwriteSizeLimit(firebaseData, "tiny");  
}  
}  
  
void loop() {  
  
}
```


Arduino

ต่อไปจะเป็นการ อธิบาย Function ในการรับส่งข้อมูล กับ Firebase

Function หลักๆของ Firebase มี 3 Function คือ

- Set คือ การเพิ่มข้อมูลลงใน Firebase แบบ(update)
- get คือ การรับข้อมูลจาก Firebase
- Push คือ การเพิ่มข้อมูลลงใน Firebase แบบSerial Monitor

Function get

```
// ----- Start of get -----  
int number = Firebase.getInt(firebaseData, "/number");  
float temp = Firebase.getFloat(firebaseData, "/temp");  
String name = Firebase.getString(firebaseData, "/name");  
bool online = Firebase.getBool(firebaseData, "isOnline");  
JsonObject list = Firebase.get(firebaseData, "list", value);  
  
firebaseData.stringData()  
// ----- END of get -----
```

Arduino

```
Function set
// ----- Start of set -----

Firebase.setInt(firebaseData"/number", 10);
Firebase.setFloat(firebaseData"/temp", 20.25);
Firebase.setString(firebaseData"/name", "IOXhop");
Firebase.setBool(firebaseData"isOnline", true);
JsonObject& objectList =
StaticJsonBuffer<200>().createObject();
objectList["autoSave"] = true;
Firebase.set(firebaseData"config", objectList);

// ----- END of set -----
```

Arduino

```
Function push
// ----- Start of push -----

Firebase.pushInt(firebaseData"/list-number", 10);
Firebase.pushFloat(firebaseData"/list-temp", 20.25);
Firebase.pushString(firebaseData"/list-name", "IOXhop");
Firebase.pushBool(firebaseData"list-isOnline", true);
JsonObject& objectList =
StaticJsonBuffer<200>().createObject();
objectList["autoSave"] = true;
Firebase.set(firebaseData"list-config", objectList);

// ----- END of push -----
```

END