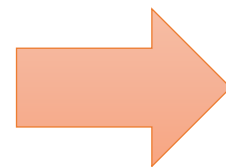


M5stackとFirebase

これから作るもの

データ
収集



データ送信

データを保存



Firebaseとは？

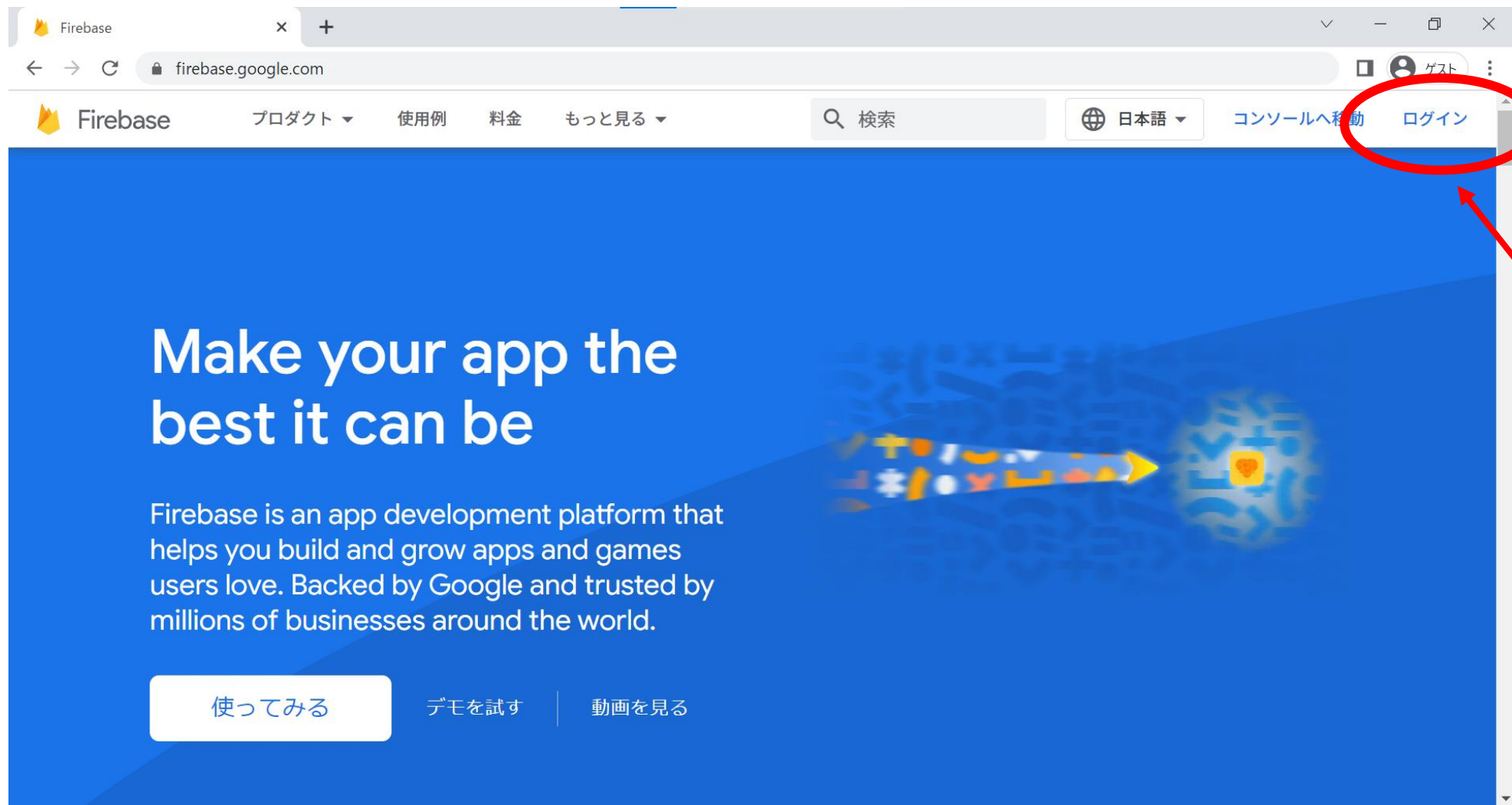
Firebaseとは、Googleが提供しているスマートフォンアプリやWebアプリケーションにおける「バックエンド開発」において、スピードの向上とコスト削減を可能にするプラットフォーム

いろいろな機能がある

機能名	内容
Firebase Realtime Database	データはJSON形式で保存され、すべてのクライアントと、ほとんどタイムラグなく同期される。
Google Analytics for Firebase	アプリの使用状況とユーザーエンゲージメントについて分析できる機能。単純なユーザーの行動だけでなく、属性別のユーザーの行動や広告の効果、課金の状況など、最大で500種類の個別のイベントを分析し、レポートを生成することも可能。
Firebase Hosting	静的なWebページやWebアプリを簡単な操作でデプロイできる機能。HTML、CSS、JavaScriptを使用する程度のWebサイトであれば、Firebase Hostingによって簡単に公開できるため、レンタルサーバーを契約したり、サーバーを立てたりする必要はありません。

などなど。。。

まずはログインから



クリック

gmailでログイン



ログイン

お客様の Google アカウントを使用

メールアドレスまたは電話番号

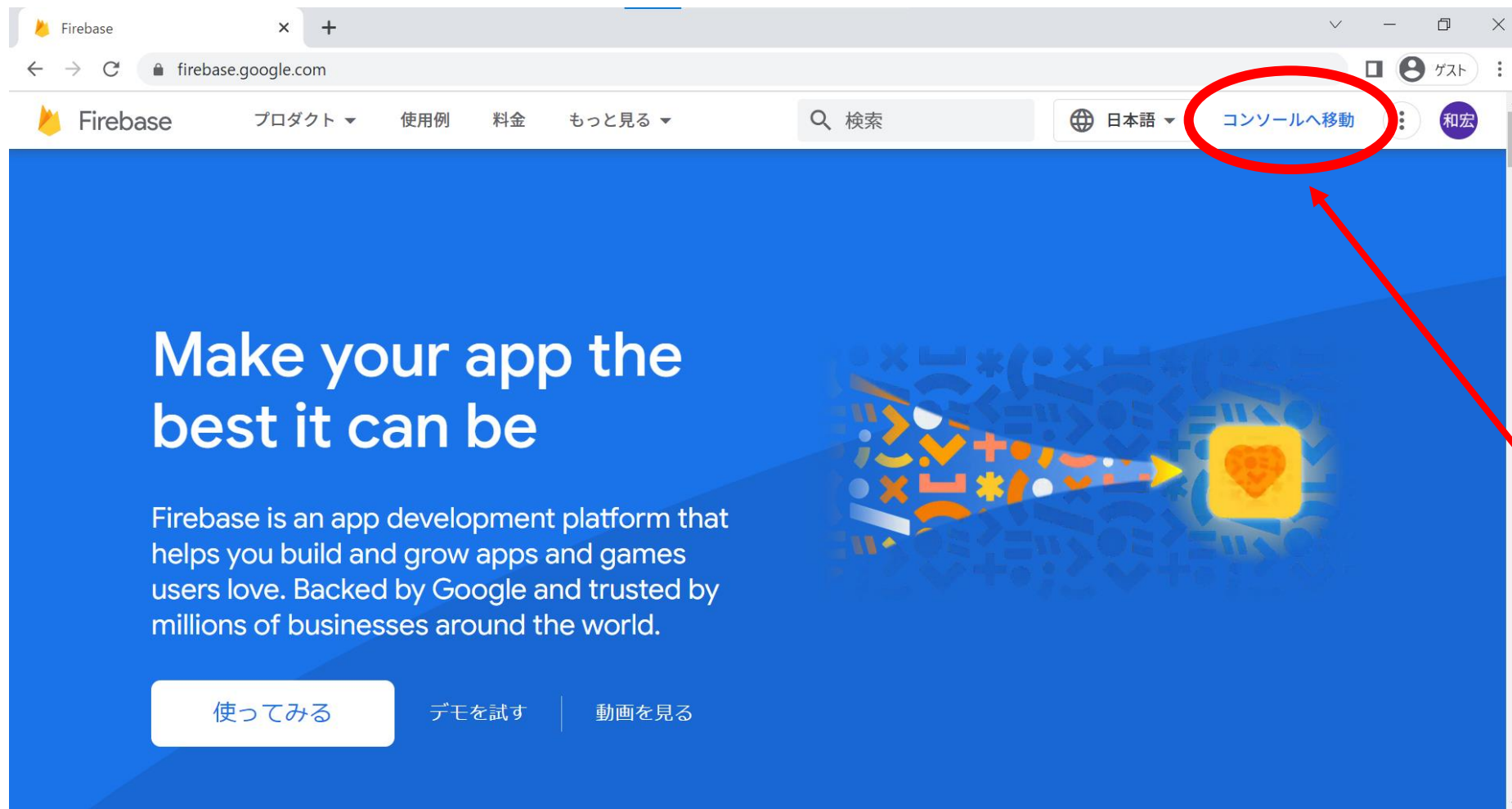
[メールアドレスを忘れた場合](#)

ご自分のパソコンでない場合は、ゲストモードを使用して非公開でログインしてください。 [詳細](#)

[アカウントを作成](#)[次へ](#)

配布したgmailアカウントで
ログインします

ログインしたら次はコンソールへ移動



クリック

プロジェクトを作成する

Firebase へようこそ

Google が提供する、アプリ インフラストラクチャの構築、アプリの品質向上、ビジネスの成長のためのツール

プロジェクトを作成

≡ ドキュメントを表示

クリック

手順1

× プロジェクトの作成（手順 1/3）

プロジェクト名

M5Database

m5database

☒ [Firebase の規約](#)に同意します

☒ 自身の取引、ビジネス、仕事、または職業のみを目的として Firebase を利用することを正式に認めます。

続行


適当なプロジェクト名をつける



チェック入れる


クリック



手順2



× プロジェクトの作成（手順 2/3）

 A/B テスト 

 Firebase プロダクト全体でのユーザー
セグメンテーションとターゲティング 

 クラッシュに遭遇していないユーザー
数 

 イベントベースの Cloud Functions ト
リガー 

 無料で無制限のレポート 

☒ このプロジェクトで Google アナリティクスを有効にする
おすすめ

前へ

続行

クリック

手順3

①日本にする

× プロジェクトの作成（手順 3/3）

Google アナリティクスの構成

アナリティクスの地域 ②

日本 ▼

データ共有の設定と Google アナリティクスの利用規約

②チェック入れる

× プロジェクトの作成（手順 3/3）

☒ Google アナリティクスデータの共有にデフォルト設定を使用します。 [詳細](#)

- × Google サービスの改善のために、アナリティクスデータを Google と共有します
- ✓ ベンチマークを有効にするために、アナリティクスデータを Google と共有します
- ✓ テクニカル サポートを有効にするために、アナリティクスデータを Google と共有します
- ✓ アナリティクスデータを Google アカウント スペシャリストと共有します

☒ [Google アナリティクス利用規約](#)に同意します。

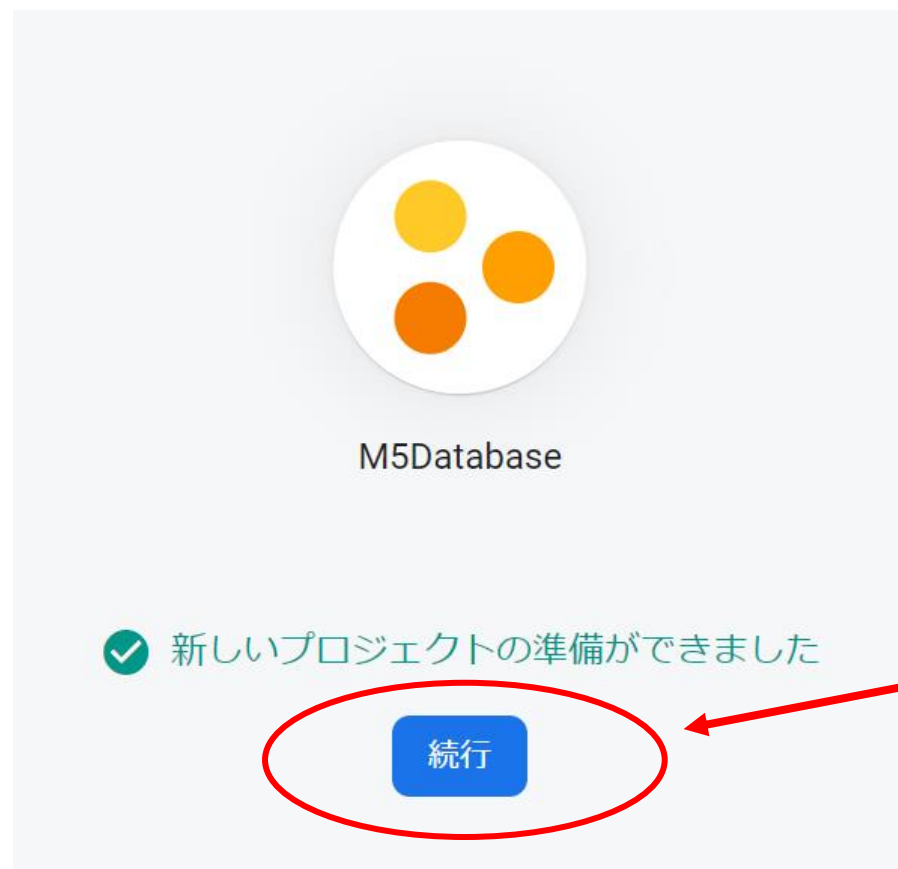
プロジェクトを作成すると、新しい Google アナリティクス プロパティが作成され、Firebase プロジェクトにリンクされます。このリンクにより、両サービスの間でデータをやり取りできるようになります。Google アナリティクスのプロパティから Firebase にエクスポートされるデータには Firebase の利用規約が適用され、Google アナリティクスにインポートされる Firebase のデータには Google アナリティクスの利用規約が適用されます。 [詳細](#)

[前へ](#)

[プロジェクトを作成](#)

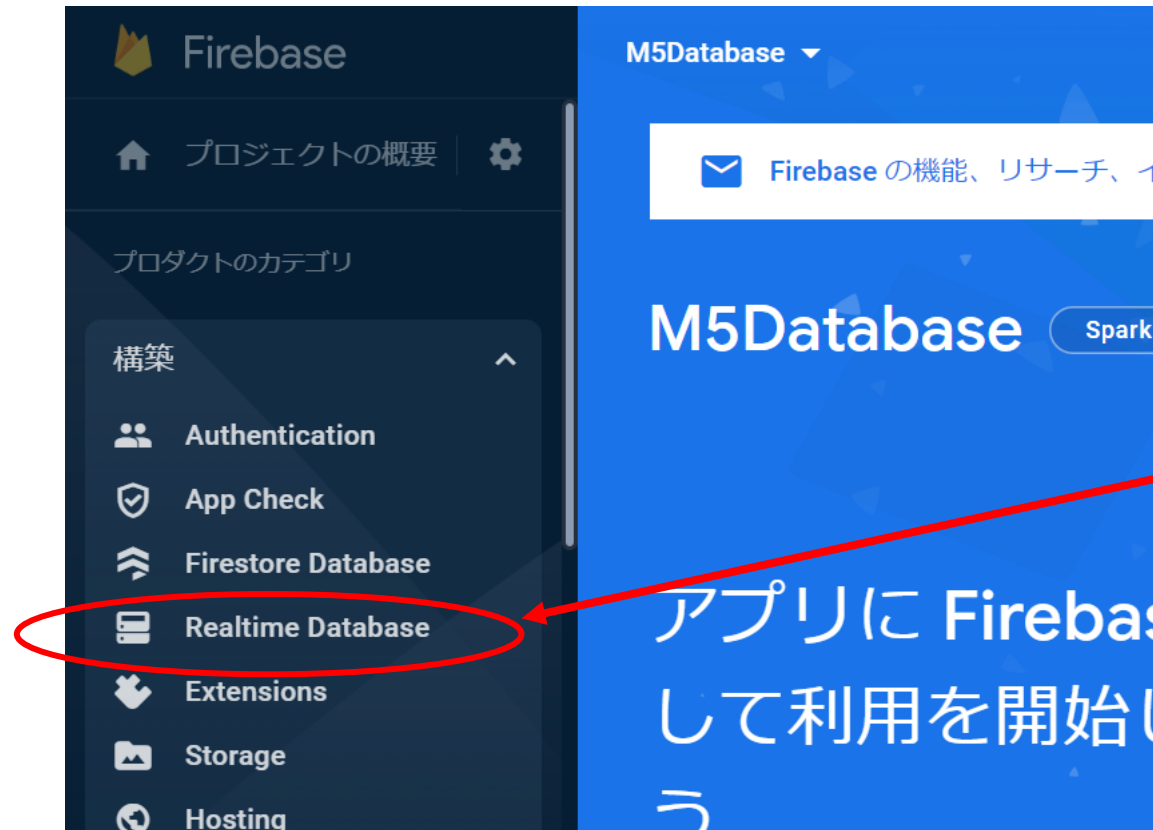
③クリック

プロジェクト作成完了です！



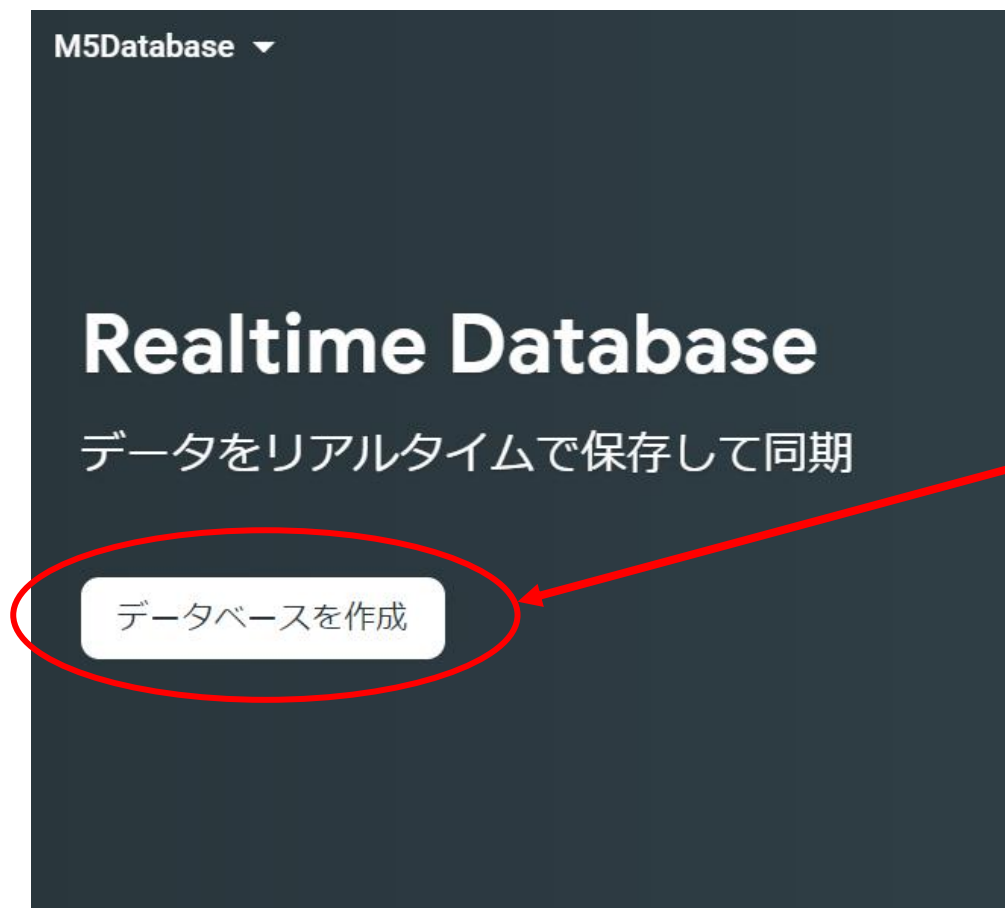
続行クリック

Realtime Databaseの構築



構築からRealtime Database
をクリック

データベースの作成



クリック

データベースの設定 1

データベースの設定



1 データベースのオプション ——— 2 セキュリティルール

ロケーションの設定は、Realtime Database データが格納される場所を定めます。

Realtime Database のロケーション

米国 (us-central1)



クリック

キャンセル

次へ

データベースの設定 2

データベースの設定

1 データベースのオプション

2 セキュリティルール

データ構造の定義後に、データのセキュリティを保護するルールを作成する必要があります。
[詳細](#)

☐ ロックモードで開始

データはデフォルトで限定公開になります。セキュリティルールで指定されているとおりに、クライアントの読み取り / 書き込み権限のみ付与されます。

☒ テストモードで開始する

データはデフォルトでオープン状態となり、迅速なセットアップが可能になります。ただし、クライアントの読み取り / 書き込み権限を長期にわたって有効にするには、30 日以内にセキュリティルールを更新する必要があります。

```
{  "rules": {    ".read": "now < 1662476400000",    // 2022-9-7    ".write": "now < 1662476400000",    // 2022-9-7  }  }
```

!

テストモードのデフォルトのセキュリティルールにより、今後 30 日間、データベース参照を所有しているユーザーなら誰でもデータベースのすべてのデータの表示、編集、削除を行うことができます

キャンセル

有効にする

①テストモードを選択

②クリック

データベースの作成が完了



The screenshot shows the 'Realtime Database' interface. At the top, there are tabs for 'データ' (Data), 'ルール' (Rules), 'バックアップ' (Backup), and '使用状況' (Usage). Below the tabs, a security warning is displayed. The main content area shows a default URL: `https://m5database-default-rtdb.firebaseio.com`, which is circled in red. Below this, a full URL with a trailing `/: null` is visible. A red arrow points from the circled URL to the Japanese text at the bottom.

Realtime Database

データ ルール バックアップ 使用状況

請求詐欺やフィッシングなどの不正行為から Realtime Database のリソースを保護します

`https://m5database-default-rtdb.firebaseio.com`

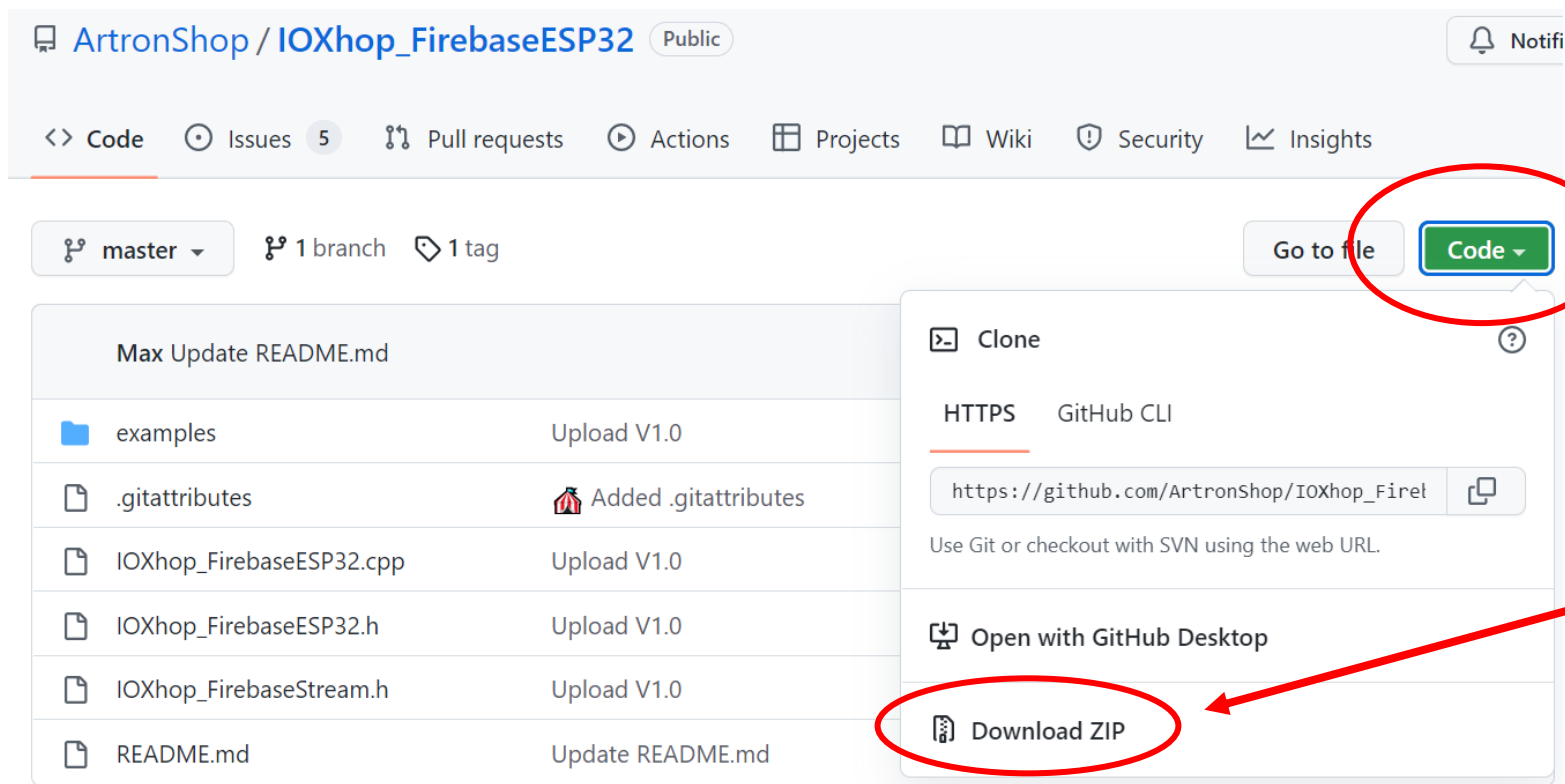
`https://m5database-default-rtdb.firebaseio.com/: null`

このアドレスは後で使います（後述）

ライブラリのインストール

下記にアクセスしライブラリをダウンロードする

https://github.com/ArtronShop/IOXhop_FirebaseESP32



①クリック

②ダウンロードする

ダウンロードしたライブラリを追加

m5firebase | Arduino 1.8.16

ファイル 編集 スケッチ ツール ヘルプ

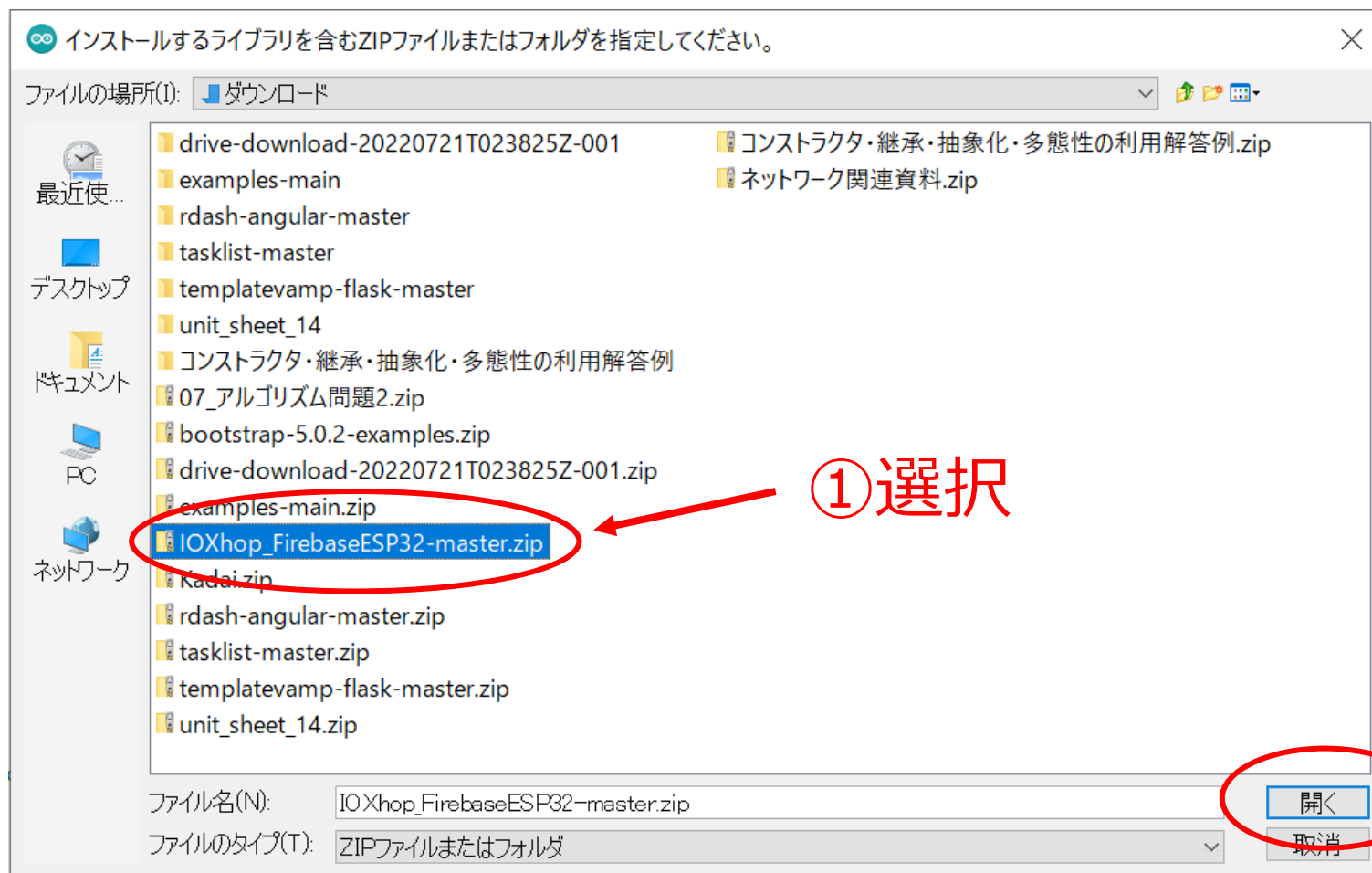
The screenshot shows the Arduino IDE interface. The 'Sketch' menu is open, and the 'Library Manager' dialog is also open. A red arrow points from the 'Sketch' menu to the 'Library Manager' dialog, and another red arrow points from the 'Library Manager' dialog to the 'ZIP形式のライブラリをインストール...' option.

検証・コンパイル Ctrl+R
マイコンボードに書き込む Ctrl+U
書込装置を使って書き込む Ctrl+Shift+U
コンパイルしたバイナリを出力 Ctrl+Alt+S
スケッチのフォルダを表示 Ctrl+K
ライブラリをインクルード
ファイルを追加...

ライブラリを管理... Ctrl+Shift+I
.ZIP形式のライブラリをインストール...
Arduino ライブラリ
Bridge
Esplora
Firmata

```
1 #incl
2
3 #incl
4 #incl
5 #incl
6 #include <WiFiGeneric.h>
7 #include <WiFiMulti.h>
8 #include <WiFiScan.h>
9 #include <WiFiServer.h>
10 #include <WiFiSTA.h>
11 #include <WiFiType.h>
```

ライブラリを選択し開く



インストール確認

m5firebase | Arduino 1.8.16

ファイル 編集 スケッチ ツール ヘルプ

The screenshot shows the Arduino IDE interface. The 'Sketch' menu is open, and the 'Include Library' option is selected. A list of libraries is displayed, with 'IOXhop_FirebaseESP32-master' circled in red. The code editor on the left shows the following code:

```
1 #include <Arduino.h>
2
3 #include <Wire.h>
4 #include <SPI.h>
5 #include <SD.h>
6 #include <WiFiGeneric.h>
7 #include <WiFiMulti.h>
8 #include <WiFiScan.h>
9 #include <WiFiServer.h>
10 #include <WiFiSTA.h>
11 #include <WiFiType.h>
12 #include <WiFiUdp.h>
13 #include "time.h"
```

ちゃんと
インストール
されました

ArduinoJsonをインストール1

m5firebase | Arduino 1.8.16

ファイル 編集 スケッチ ツール ヘルプ

The screenshot shows the Arduino IDE interface. The 'Sketch' menu is open, and the 'Include Library' option is selected. A red arrow points from this option to the 'Library Manager' window, which is also open. In the 'Library Manager' window, the 'Library Manager...' option is selected, and a red arrow points from it to the 'Install Library from ZIP File...' option. The 'Library Manager' window also shows the 'Arduino Library' and 'Bridge' options.

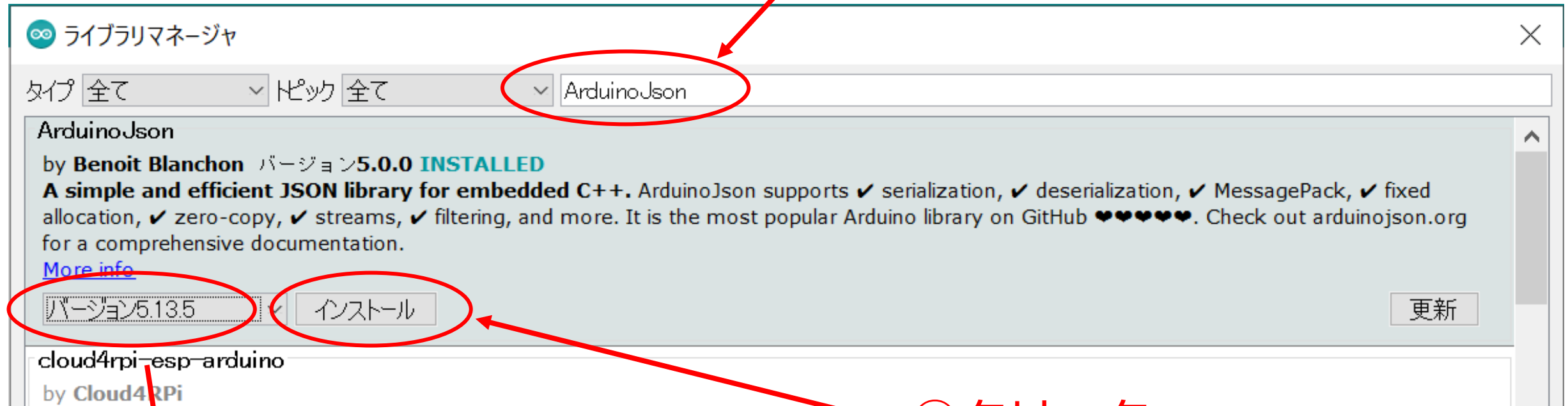
検証・コンパイル Ctrl+R
マイコンボードに書き込む Ctrl+U
書込装置を使って書き込む Ctrl+Shift+U
コンパイルしたバイナリを出力 Ctrl+Alt+S
スケッチのフォルダを表示 Ctrl+K
ライブラリをインクルード
ファイルを追加...

ライブラリを管理... Ctrl+Shift+I
.ZIP形式のライブラリをインストール...
Arduino ライブラリ
Bridge

```
1 #incl
2
3 #incl
4 #incl
5 #incl
6 #include <WiFiGeneric.h>
7 #include <WiFiMulti.h>
8 #include <WiFiScan.h>
9 #include <WiFiServer.h>
10 #include <WiFiClient.h>
```

ArduinoJsonをインストール2

① ArduinoJsonで検索



② バージョン5.13.5を選択

最新の6系でインストールしても動かないので注意！

③ クリック

インストール確認

m5firebase | Arduino 1.8.16

ファイル 編集 スケッチ ツール ヘルプ

The screenshot shows the Arduino IDE interface. The 'Sketch' menu is open, displaying options: '検証・コンパイル' (Ctrl+R), 'マイコンボードに書き込む' (Ctrl+U), '書込装置を使って書き込む' (Ctrl+Shift+U), 'コンパイルしたバイナリを出力' (Ctrl+Alt+S), 'スケッチのフォルダを表示' (Ctrl+K), 'ライブラリをインクルード' (highlighted), and 'ファイルを追加...'. The 'Libraries' panel on the right shows a list of installed libraries: 'TFT', 'Temboo', '提供されたライブラリ' (Provided Libraries), 'ArduinoJson' (highlighted with a red circle), and 'ArduinoOTA'. A red arrow points from the Japanese text on the right to the 'ArduinoJson' library.

検証・コンパイル Ctrl+R
マイコンボードに書き込む Ctrl+U
書込装置を使って書き込む Ctrl+Shift+U
コンパイルしたバイナリを出力 Ctrl+Alt+S
スケッチのフォルダを表示 Ctrl+K
ライブラリをインクルード
ファイルを追加...

m5firebase

```
1 #incl
2
3 #incl
4 #incl
5 #incl
6 #include <WiFiGeneric.h>
7 #include <WiFiMulti.h>
8 #include <WiFiScan.h>
9 #include <WiFiServer.h>
10 #include <WiFiSTA.h>
```

TFT
Temboo
提供されたライブラリ
ArduinoJson
ArduinoOTA

ちゃんと
インストール
されました

サンプルプログラム


M5_firebase.ino を開きます

名前

 M5_firebase.ino

 M5toFirebase.cpp

 M5toFirebase.h

 WifiConnect.cpp

 WifiConnect.h

← Firebase送受信用ソースファイル

← WiFi接続用ソースファイル

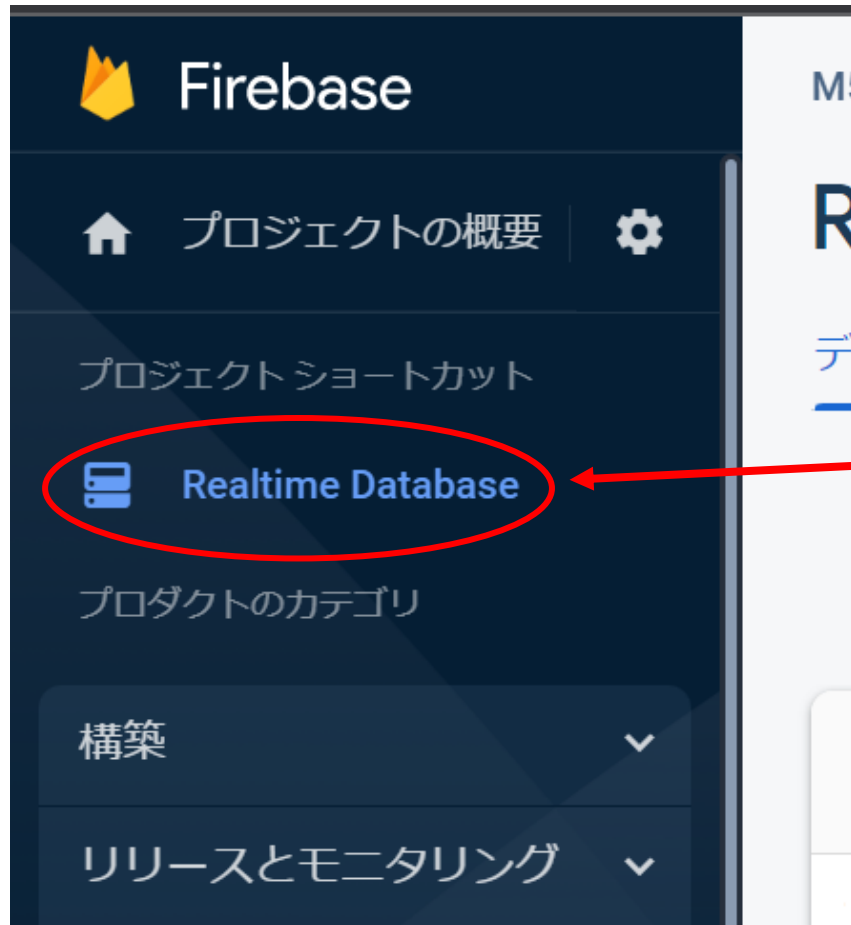
以下を書き換える

M5_firebase	M5toFirebase.cpp	M5toFirebase.h	WifiConnect.cpp	WifiConnect.h
-------------	------------------	----------------	-----------------	---------------

```
1 /* インクルードガード */
2 #pragma once
3
4 #define FIREBASE_DATABASE_URL "https://を除いたfirebaseのURLを貼り付け"
5
6 class M5toFirebase // classの定義
7 {
8     private :           // privateはクラス内からしかアクセスできない
9
10    public :              // publicはどこからでもアクセス可能
11        void init();
12        void setData(String path, float t, float h);
13};
```

Realtime Databaseにアクセスし、コピーしてくる

Firestoreにアクセス



作ったRealtime Databaseは
左のフレームからアクセスできます

Firestoreのアドレスをコピー

Realtime Database

[データ](#)[ルール](#)[バックアップ](#)[使用状況](#)

 請求詐欺やフィッシングなどの不正行為から Realtime Database のリソースを保護します [AI](#)

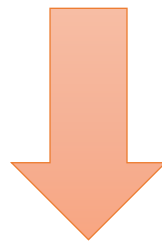
 <https://m5database-default-rtdb.firebaseio.com>

<https://m5database-default-rtdb.firebaseio.com/>: null

これをクリックし右のアドレスをコピー

Firestoreのアドレスを入力

<https://m5database-default-rtdb.firebaseio.com/>



[https://](https://m5database-default-rtdb.firebaseio.com/) を除いたものを入力

M5_firebase

M5toFirebase.cpp

M5toFirebase.h §

WifiConnect.cpp

WifiConnect.h

```
1 /* インクルードガード */
2 #pragma once
3
4 #define FIREBASE_DATABASE_URL "m5database-default-rtdb.firebaseio.com/"
5
6 class M5toFirebase // classの定義
7 {
```

まずはWiFi設定

M5_firebase.ino

```
9 void setup() {  
10   M5.begin();  
11   wificonnect.init();  
12   m5fb.init();  
13 }
```



WifiConnect.cpp

```
// 初期化处理  
15 void WifiConnect::init(){  
16  
17   // Wi-Fi接続  
18   WiFi.begin(WIFI_SSID, WIFI_PASSWORD);  
19   Serial.print("connecting");  
20   while(WiFi.status() != WL_CONNECTED) {  
21     Serial.print(".");  
22     delay(500);  
23   }  
24   Serial.println();  
25  
26   // WiFi Connected  
27   Serial.println("\nWiFi Connected.");  
28   Serial.println(WiFi.localIP());  
29  
30 }
```

WiFi接続開始

WiFi接続されるまで待つ

local IPの表示

Firebaseへの接続

M5_firebase.ino

```
9 void setup() {  
10   M5.begin();  
11   wificonnect.init();  
12   m5fb.init();
```

M5toFirebase.cpp

```
7 // 初期化处理  
8 void M5toFirebase::init() {  
9   // Firebase初期化  
10   Firebase.begin(FIREBASE_DATABASE_URL);  
11 }
```

→ Firebaseと接続開始

Firestoreに送信

```
if (M5.BtnC.wasPressed())
```

```
{
```

```
float temperature = 30;
```

```
float humidity = 65;
```

```
String strpass = "/M5Stack";
```

```
m5fb.setData(strpass, temperature, humidity);
```

```
}
```

temperatureに30をセット

humidityに65をセット

/M5Stackオブジェクトに
temperature、humidityを格納

setData関数の中身

M5_firebase.ino

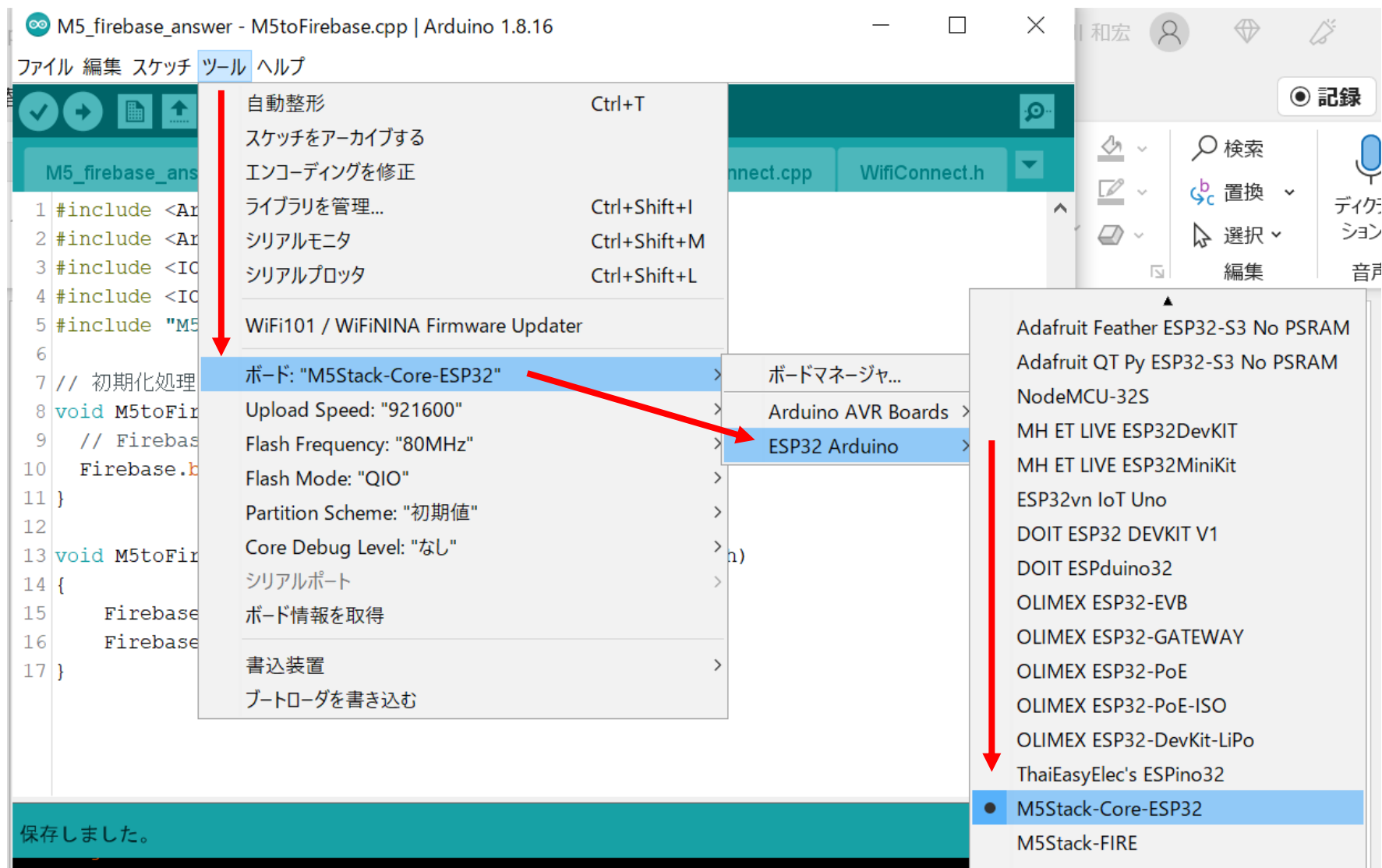


```
m5fb.setData(strpass, temperature, humidity);
```

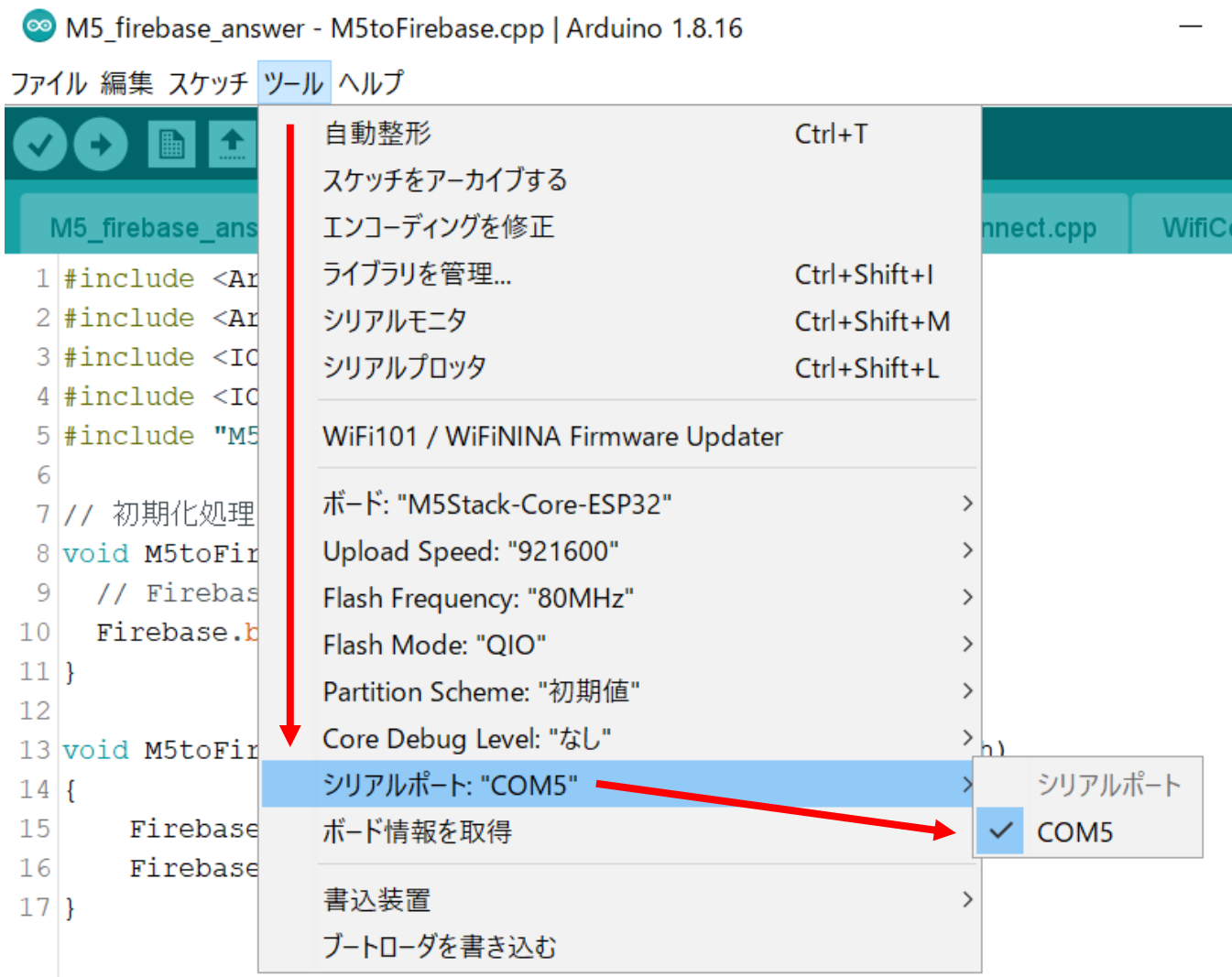
```
13 void M5toFirebase::setData(String path, float t, float h)
14 {
15     Firebase.setFloat(path + "/temperature", t);
16     Firebase.setFloat(path + "/humidity", h);
17 }
```

M5toFirebase.cpp

コンパイルする前に設定 1



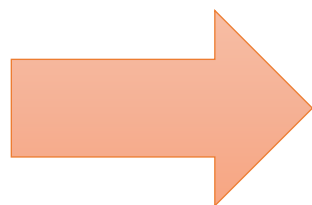
コンパイルする前に設定 2



コンパイルして実行すると



右ボタンクリック



M5Database ▼

Realtime Database

データ ルール バックアップ 使用状況

請求詐欺やフィッシングなどの不正行為から Realtime Database のリソース

<https://m5database-default-rtdb.firebaseio.com>

<https://m5database-default-rtdb.firebaseio.com/>

▼ M5Stack

humidity: 65

temperature: 30

ちゃんと指定した通りに値が格納されました！

setInt以外の関数について

下記にアクセスする

https://github.com/ArtronShop/IOXhop_FirebaseESP32

下の方に行くとまずgetter関数の説明があります

Features

Get

Read value from object in firebase. And support data type `int` `float` `String` `bool` and `JsonObject` via function `type get[type](String path)` or `void get(String path, [type] &value)`

```
int getInt(String path) ;
float getFloat(String path) ;
String getString(String path) ;
bool getBool(String path) ;
void get(String path, int &value) ;
void get(String path, float &value) ;
void get(String path, String &value) ;
void get(String path, bool &value) ;
JsonVariant get(String path) ;
```

getter関数について

Realtime Databaseにある値をgetすることもできます

戻り値で
取得する
タイプ

引数のアド
レス渡しで
取得するタ
イプ

```
int getInt(String path) ;  
float getFloat(String path) ;  
String getString(String path) ;  
bool getBool(String path) ;  
void get(String path, int &value) ;  
void get(String path, float &value) ;  
void get(String path, String &value) ;  
void get(String path, bool &value) ;  
JsonVariant get(String path) ;
```

int型の値を取得するとき

float型の値を取得するとき

String型の値を取得するとき

bool型の値を取得するとき

こちらがsetter関数

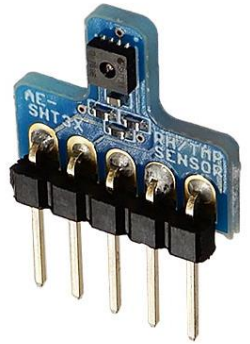
int型だけでなく、float、String、Bool型も

Realtime Databaseに格納できます

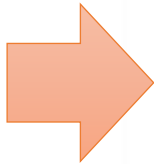
```
void setInt(String path, int value) ;  
void setFloat(String path, float value, int point = 2) ;  
void setString(String path, String value) ;  
void setBool(String path, bool value) ;  
void set(String path, int value) ;  
void set(String path, float value, int point = 2) ;  
void set(String path, String value) ;  
void set(String path, bool value) ;  
void set(String path, JsonVariant value) ;
```

ここからは温湿度センサの値を取得します

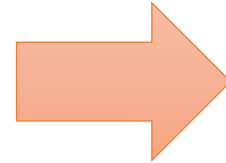
データを保存



温湿度
データ
収集



M5STACK



データ送信



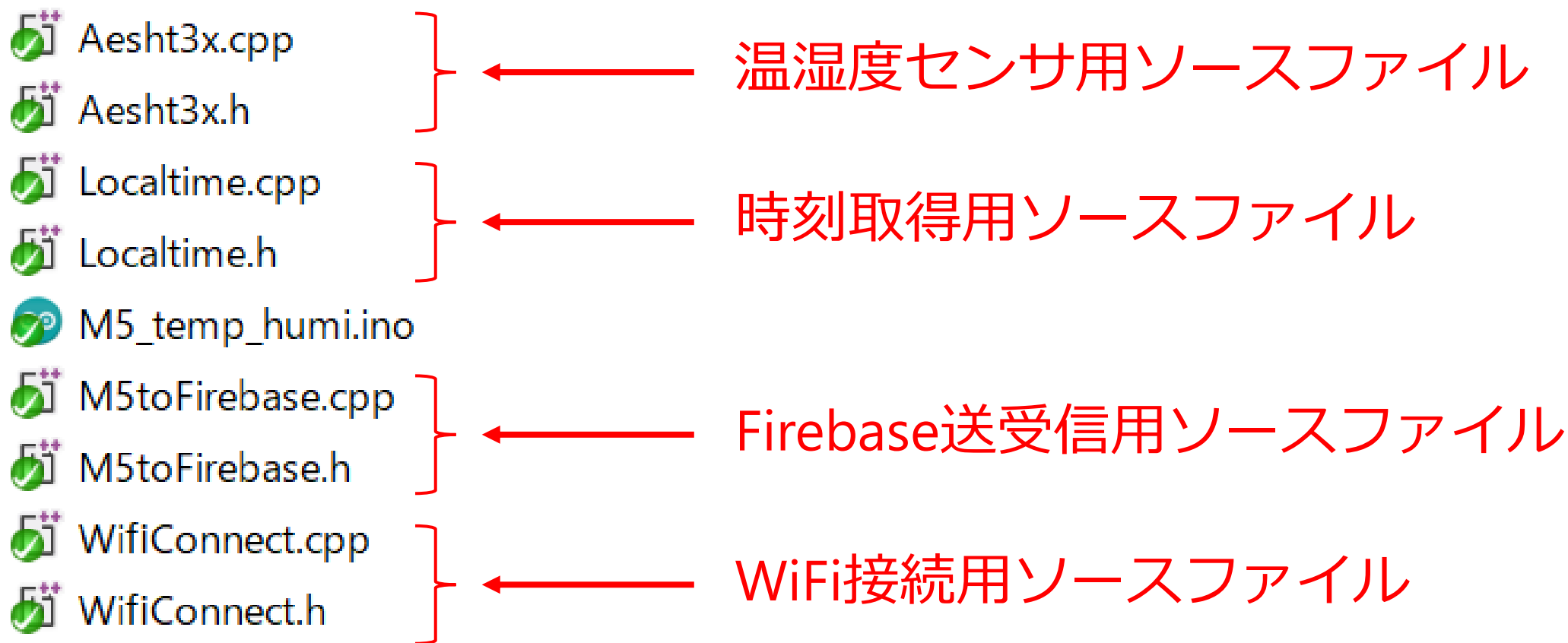
Firebase
Realtime
Database

M5Stackとの接続

温湿度センサ (AE-SHT3X)	M5Stack
GND	G (下側)
SCL	SCL (下側)
SDA	SDA (下側)
+V	5V(下側)

サンプルプログラム

M5_temp_humi.ino を開きます



ソースの簡単な説明1

```
13 void setup() {  
14     M5.begin();  
15     wificonnect.init();  
16     m5fb.init();  
17  
18     M5.Lcd.setTextSize(2);  
19     M5.Lcd.setCursor(0, 0);  
20     M5.Lcd.println("Click the right button to send data");  
21  
22     sht3x.init();  
23     lcltime.init();  
24 }
```

Wi-Fi接続開始

Firebaseと接続開始

画面表示

温湿度センサの初期設定 (I2C)

日時取得の初期設定 (ntpサーバー)

ソースの簡単な説明2

```
37  if (M5.BtnC.wasPressed())
38  {
39      M5.Lcd.clear();
40      M5.Lcd.setTextSize(2);
41      M5.Lcd.setCursor(0, 0);
42      M5.Lcd.println("Click the right button to send data");
43
44      float temperature = 0;
45      float humidity = 0;
46
47      sht3x.getTempHumi(&temperature, &humidity);
48
```

} 画面クリア

ソースの簡単な説明3

```
49     String strpass = "";
50     char pathtime[64];
51     lcltime.getTimePath(pathtime);
52     strpass = pathtime;
53
54     m5fb.setData(strpass, temperature, humidity);
55
56     M5.Lcd.setCursor(0, 100);
57     M5.Lcd.println("Path : " + strpass);
58     M5.Lcd.print("Temperature : ");
59     M5.Lcd.setTextColor(ORANGE);
60     M5.Lcd.println(String(temperature) + " deg");
61     M5.Lcd.setTextColor(WHITE);
62     M5.Lcd.print("Humidity : ");
63     M5.Lcd.setTextColor(CYAN);
64     M5.Lcd.println(String(humidity) + " percent");
65     M5.Lcd.setTextColor(WHITE);
```

画面表示

コンパイルして書き込み



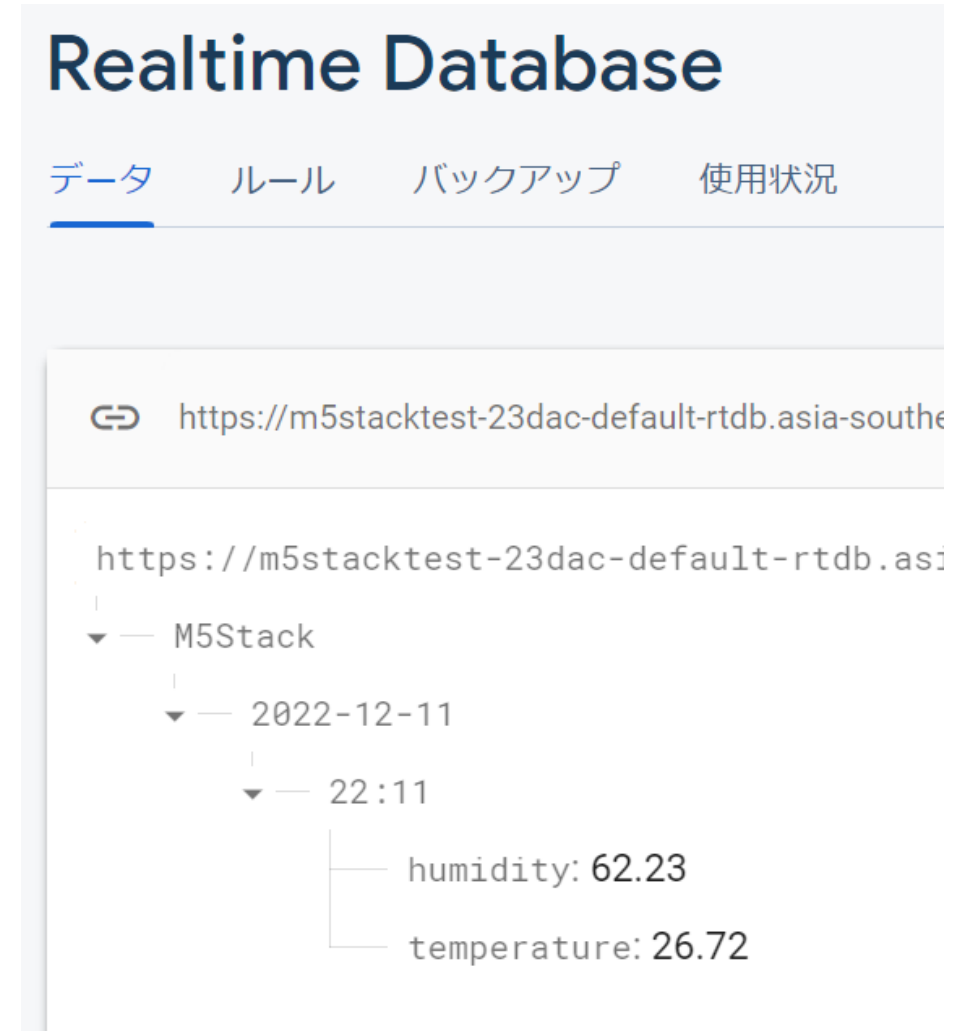
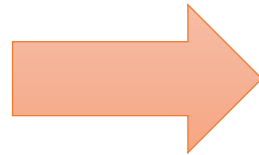
クリック

```
17 |  
18 | M5.Lcd.setTextSize(2);  
ボードへの書き込みが完了しました。  
Writing at 0x000dbce9... (89 %)  
Writing at 0x000e0e3d... (91 %)  
Writing at 0x000e6bd4... (94 %)  
Writing at 0x000ebf21... (97 %)  
Writing at 0x000f183a... (100 %)  
Wrote 935088 bytes (597794 compressed) at 0x00010000 in  
Hash of data verified.  
Leaving...  
Hard resetting via RTS pin...
```

Realtime Databaseに温湿度が格納されました

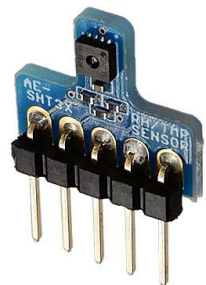


M5Stackの
右ボタンを
押すと、、、



LEDのONOFFをクラウドを介して切り替えます

データを保存



温湿度データ収集



M5STACK

データ送信



Firebase
Realtime
Database



データの変更
を受信

LED



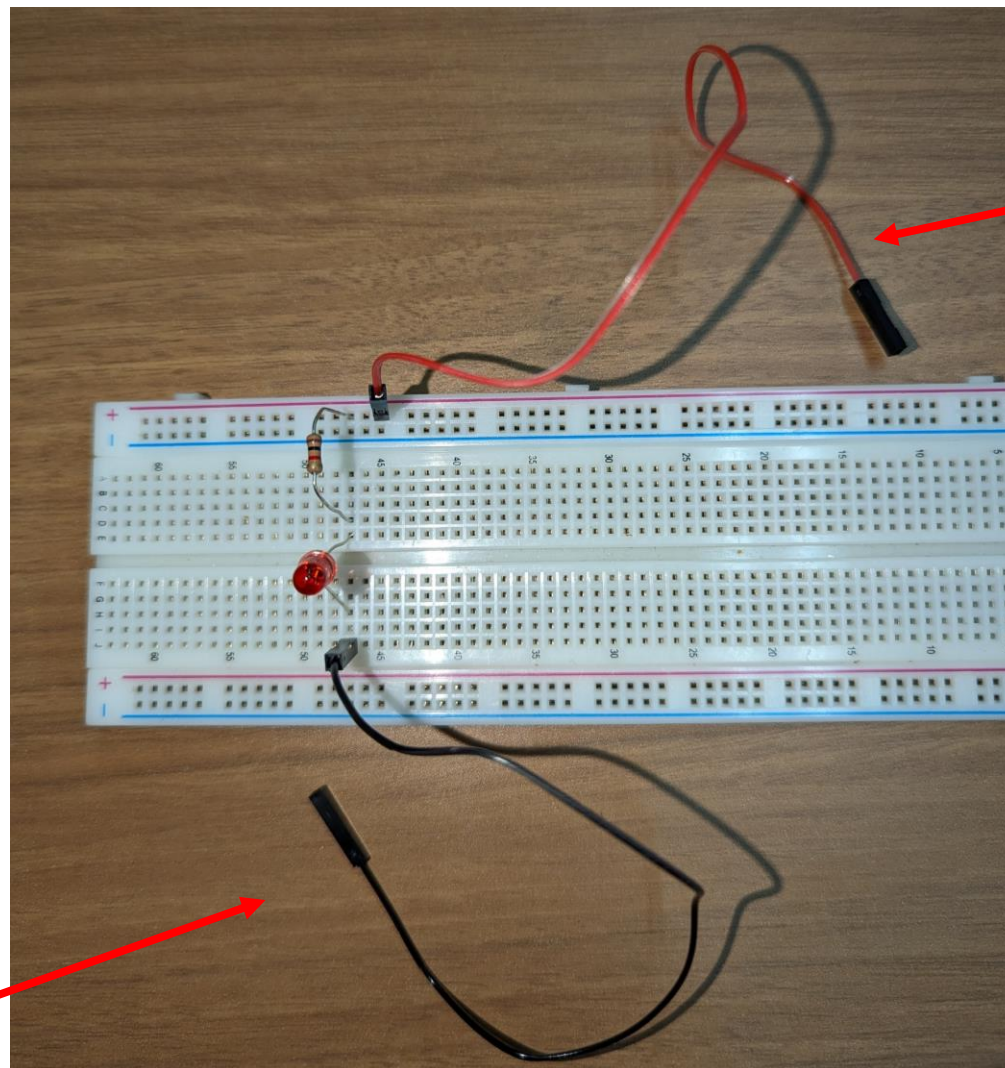
点灯・消灯

M5Stackとの接続

温湿度センサ (AE-SHT3X)	M5Stack
GND	G (下側)
SCL	SCL (下側)
SDA	SDA (下側)
+V	5V(下側)

LED	M5Stack
アノード側	R2 (左側)
カソード側	G (左側)

配線図



赤のジャンパ線をM5Stackの
左側R2ピンへ

黒のジャンパ線をM5Stackの
左側Gピンへ

サンプルプログラム

M5_temp_humi_led.ino を開きます

📄 Aesht3x.cpp

📄 Aesht3x.h

📄 Localtime.cpp

📄 Localtime.h

🔌 M5_temp_humi_led

📄 M5toFirebase.cpp

📄 M5toFirebase.h

📄 WifiConnect.cpp

📄 WifiConnect.h

← 温湿度センサ用ソースファイル

← 時刻取得用ソースファイル

← Firestore送受信用ソースファイル
(変更有)

← WiFi接続用ソースファイル

Realtime DatabaseにLED制御用keyを作成1

 <https://m5stacktest-23dac-default-rtdb.asia-southeast1.firebaseio.com>

<https://m5stacktest-23dac-default-rtdb.asia-southeast1.firebaseio.com/> null



クリック

Realtime DatabaseにLED制御用keyを作成2



The screenshot shows the Firebase Realtime Database console interface. At the top, the database URL is displayed: `https://m5stacktest-23dac-default-rtdb.asia-southeast1.firebaseio.com`. Below this, the path `/M5Stack/led` is entered into the input field. The input field is highlighted with a red border. To the right of the input field are buttons for '値' (Value), '+', and '×'. Below the input field are buttons for 'キャンセル' (Cancel) and '追加' (Add). A red arrow points from the text '入力' (Input) to the '追加' button.

`https://m5stacktest-23dac-default-rtdb.asia-southeast1.firebaseio.com`

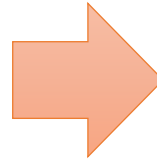
`https://m5stacktest-23dac-default-rtdb.asia-southeast1.firebaseio.com/`

`/M5Stack/led` 値 + ×

キャンセル 追加

入力

Realtime DatabaseにLED制御用keyを作成3



Realtime DatabaseにLED制御用keyを作成4



https://m5stacktest-23dac-default-rtdb.asia

https://m5stacktest-23dac-default-rtdb.asia

/M5Stack/led false

キャンセル 追加

falseを入力して
追加をクリック

Realtime DatabaseにLED制御用keyを作成5

 <https://m5stacktest-23d.firebaseio.com>

<https://m5stacktest-23d.firebaseio.com>

▼ — M5Stack

└─ led: false

/M5Stack/led の値が
falseで追加されました

Realtime Databaseにあるデータを取得

M5toFirebase.cpp

```
9 // 初期化处理
10 void M5toFirebase::init() {
11     // Firebase初期化
12     Firebase.begin(FIREBASE_DATABASE_URL);
13     digitalWrite(LED_PIN, Firebase.getBool("/M5Stack/led"));
14 }
```

/M5Stack/led の値が
falseならLED消灯
trueならLED点灯

非同期でクラウド上の変更を確認する

M5StackのCPUであるESP32では

FreeRTOSの一部のAPIが使用可能



クラウド上の変更確認用タスクを作成

```
22 void M5toFirebase::startFirebaseStream()
23 {
24     Firebase.stream("", [](FirebaseStream stream) {
25         if (stream.getEvent() == "put" && stream.getPath() == "/led") {
26             digitalWrite(LED_PIN, stream.getDataBool());
27         }
28     });
29 }
```

Realtime Databaseの値を変更1



クリックしてfalseから
trueに書き換えてEnter

Realtime Databaseの値を変更2

 <https://m5stacktest-23dac-default.firebaseio.com>

<https://m5stacktest-23dac-default.firebaseio.com>



LEDが点灯していれば成功です