



**G's ACADEMY**  
**TOKYO**

# Firebase

Googleアカウントが必要です



# アジェンダ

---

❖ オブジェクト変数

FireBase RealTimeDB

課題実習タイム (Chatアプリ or リアルタイム通信)

# オブジェクト

- Object -

# JavaScriptのオブジェクト

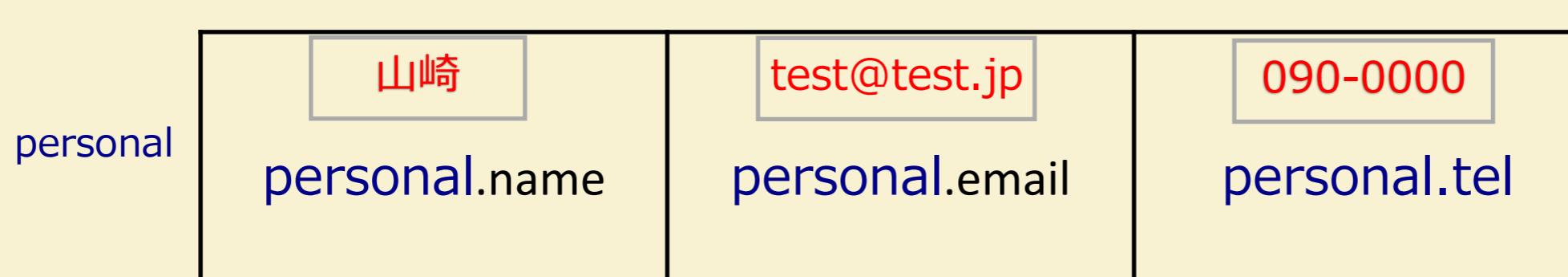
## 【 オブジェクト (Object) 】

オブジェクトは配列と違い「インデックス」ではなく「プロパティ」で値を管理することができます。プロパティは「名前・値」のペアになっており最近ではよく使用されるデータ保持の方法の1つです。

```
<script>
const personal = { name:"山崎", email:"test@test.jp", tel:"090-0000" };
const personals = {
    per1: { name:"山崎", email:"test1@test.jp", tel:"090-1111" },
    per2: { name:"鈴木", email:"test2@test.jp", tel:"090-2222" }
};
</script>
```

## 【 オブジェクトの参照イメージ】

personalオブジェクト内のプロパティに値が格納されます。personalの中の値を取得する方法は、「`personal.プロパティ名`」で取得可能



# 自作チャット作成方法

## 初級編



# Key取得

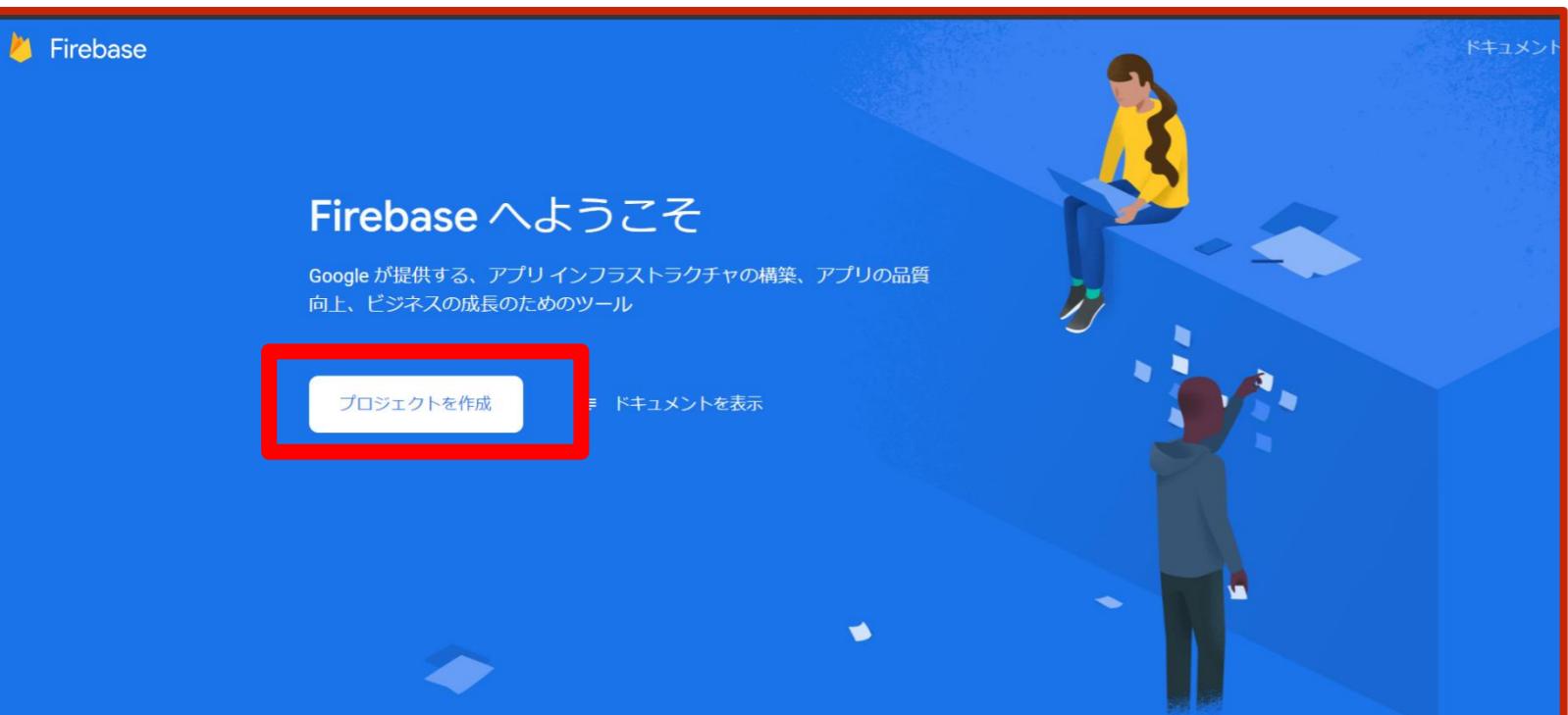
Chromeブラウザでアクセス

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

1

The screenshot shows the official Firebase website homepage. At the top, there is a navigation bar with links for 'Firebase', 'プロダクト', '使用例', '料金', 'ドキュメント', and 'サポート'. To the right of the navigation bar are a search bar, a 'Language' dropdown, and a red-bordered 'コンソールへ移動' (Move to Console) button. A large, stylized illustration of two people working on a laptop and a smartphone is centered on the page. On the left side, there is Japanese text: 'Firebase で モバイルとウェブ アプリを成功させる'. Below this text are two buttons: '使ってみる' (Try it) and '動画を見る' (Watch video). The entire page has a blue background.

# 1.新規プロジェクト作成



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

初回画面

デモプロジェクトを見る iOS

Firebase プロジェクトのコンテナ  
プロジェクト内の Database や Analytics を有しています。[Learn more](#)

ドキュメントに移動

最近のプロジェクト

+ プロジェクトを追加 デモプロジェクトを見る

すべての Firebase プロジェクト

2回目以降

The screenshot shows the initial Firebase console interface. A red box highlights the 'Create Project' button at the top left. Below it, there's a section for a demo project with an 'iOS' badge. To the right, there's a summary of recent projects and a prominent 'Add Project' button. The bottom part of the screen shows a grid of recent projects with blurred names. Red boxes highlight the 'Create Project' button in the first section and the 'Add Project' button in the second section.

× プロジェクトの作成 (手順 1/3)

まず  
プロジェクトに名前をつける②

1

プロジェクト名

GSdemo

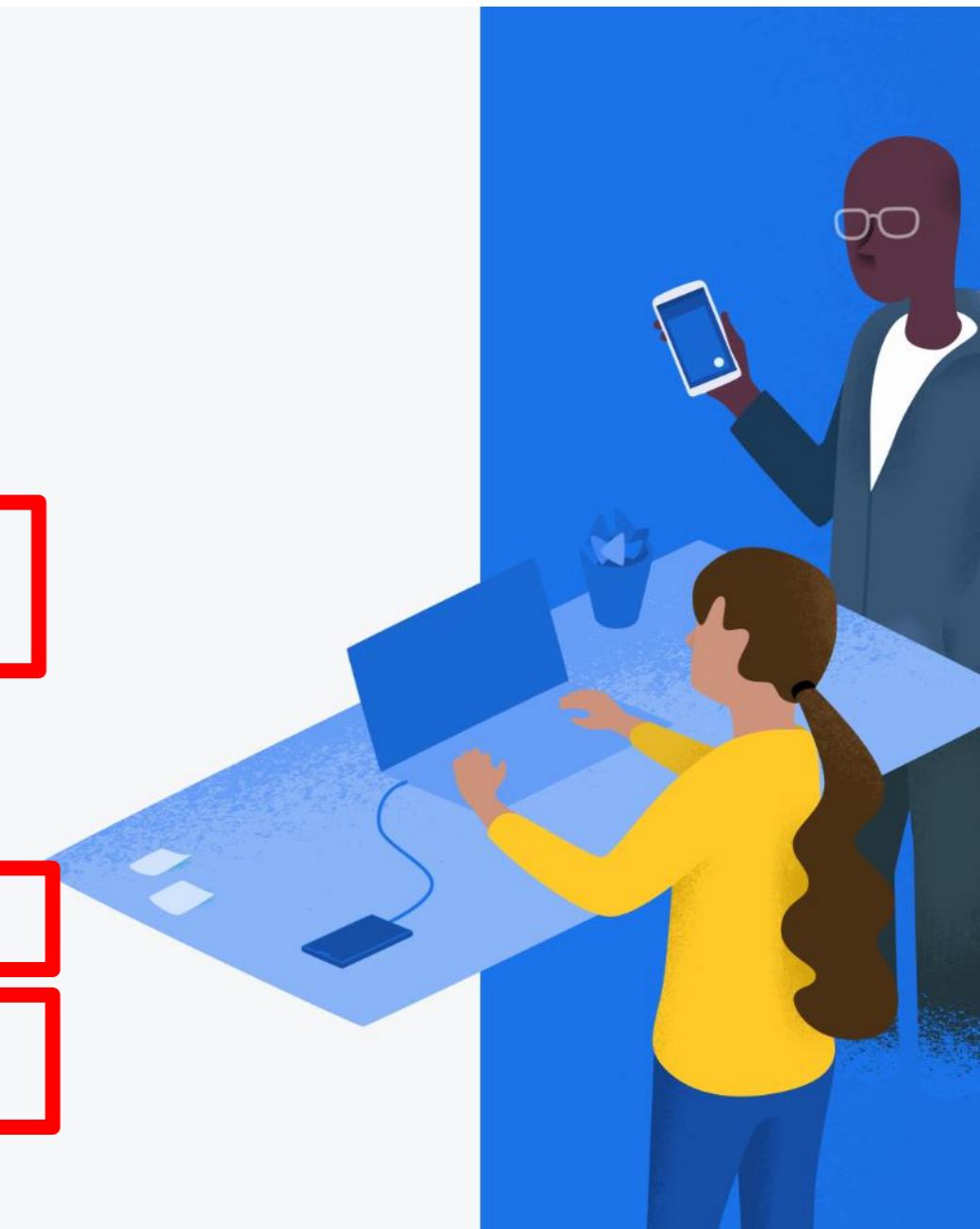
gsdemo-29f17

2

[Firebase の規約](#)に同意します

3

続行



× プロジェクトの作成（手順 2/2）

## Google アナリティクス (Firebase プロジェクト向け)

Google アナリティクスは無料かつ無制限のアナリティクスソリューションで、Firebase Crashlytics、Cloud Messaging、アプリ内メッセージング、Remote Config、A/B Testing、Predictions、Cloud Functions で、ターゲティングやレポートなどが可能になります。

Google アナリティクスにより、以下の機能が有効になります。

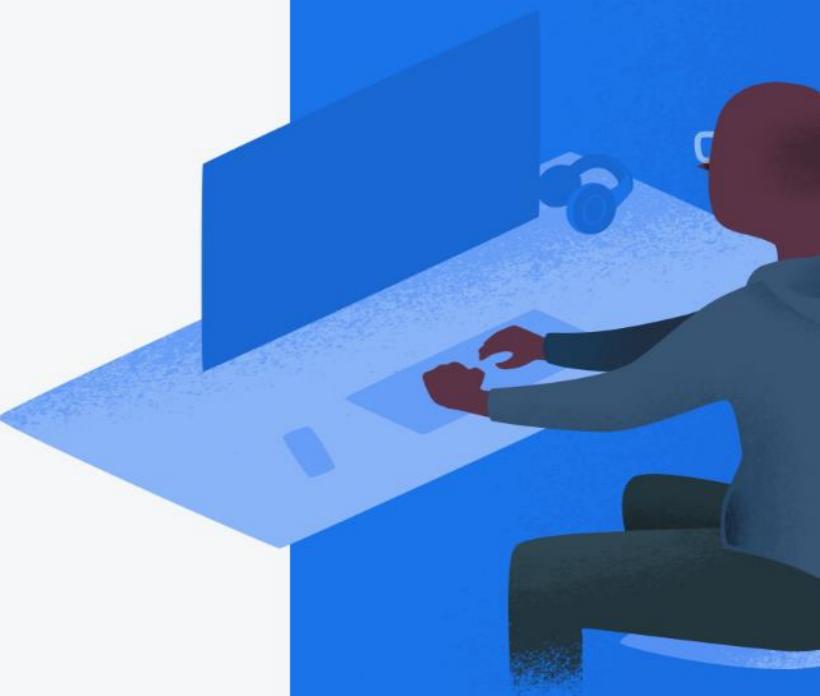
- × A/B テスト ②
- × Firebase プロダクト全体でのユーザー登録とターゲティング ②
- × ユーザー行動の予測 ②
- × クラッシュに遭遇していないユーザー ②
- × イベントベースの Cloud Functions トリガー ②
- × 無料で無制限のレポート ②

1

このプロジェクトで Google アナリティクスを有効にする  
推奨

2

プロジェクトを作成



前へ



gsdemo

✓ 新しいプロジェクトの準備ができました

続行



# 5. 「</>」を選択（Webアプリ）

The screenshot shows the Firebase Project Overview page for a project named 'dev12'. The left sidebar contains navigation links for 'Authentication', 'Database', 'Storage', 'Hosting', 'Functions', 'ML Kit', 'Crashlytics, Performance, Test L...', 'Analytics', 'Dashboard, Events, Conversions,...', and 'Predictions'. The main content area displays a blue background with white text: 'dev12 Spark プラン' at the top, followed by 'アプリに Firebase を追加して利用を開始しましょう' in large font, and 'ほとんどの Firebase 機能と、iOS や Android アプリ用のアナリティクスが含まれた Core SDK をインストールしてください' below it. At the bottom, there are three circular icons for 'iOS', 'Android', and 'Web' (represented by '</>'), with the '</>' icon highlighted by a red square box. A woman in a yellow sweater is shown interacting with a screen, and another person is visible in the background holding a smartphone.

# 6. アプリ名「chat」と登録。

× ウェブアプリに Firebase を追加

1 アプリの登録

アプリのニックネーム ②

chat

今日はChatにしておく

このアプリの **Firebase Hosting** も設定します。 [詳細](#)

Hostingは後で設定することもできます。いつでも無料で始めることができます。

アプリを登録

2 Firebase SDK の追加



# 7. コピーボタンで[CODE]をコピーしておく。

× Firebaseに接続する大事にKEYになります。

✓ アプリの登録

2 Firebase SDK の追加

これらのスクリプトをコピーして `<body>` タグの下部に貼り付けてください。この作業は Firebase サーバーを使用する前に行ってください。

1. 以下コードをコピー

```
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/6.0.2.firebaseio-app.js"></script>

<!-- TODO: Add SDKs for Firebase products that you want to use
      https://firebase.google.com/docs/web/setup#config-web-app -->

<script>
  // Your web app's Firebase configuration
  var firebaseConfig = {
    apiKey: "AIzaSyC8UmHcHFC4S4eDgCSbjzYJZXdyoVZZ6x8",
    authDomain: "chats-e9be4.firebaseio.com",
    databaseURL: "https://chats-e9be4.firebaseio.com",
    projectId: "chats-e9be4",
    storageBucket: "chats-e9be4.appspot.com",
    messagingSenderId: "1013612958152",
    appId: "1:1013612958152:web:29d273406d53cc81"
  };
  // Initialize Firebase
  firebase.initializeApp(firebaseConfig);
</script>
```

ウェブ向け Firebase の詳細については、こちらをご覧ください: [使ってみる](#)、[ウェブ SDK API リファレンス](#)、[サンプル](#)

2. コンソールに進む



## 8. エディターを開きスクリプトを記述する準備をしましょう。

```
1  <!DOCTYPE html>
2 ▼ <html lang="ja">
3 ▼ <head>
4   .... <meta charset="UTF-8">
5   .... <title>Document</title>
6 </head>
7 ▼ <body>
8   ....
```

```
9   ....
```

最低限のHTMLは記述しておきました。

```
11 ...
12
13 </body>
14 </html>
```

9. コピーしたスクリプトを「貼り付け」ます。  
場所は「</body>」の上にします。

simple.html

```
22
23  <!-- JQuery -->
24  <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
25  <!-- JQuery -->
26
27
28  <!!--** 以下Firebase **-->
29
30
31
32
33  Firebaseに接続する大事なKEYをここに貼り付けます
34
35
36
37
```

# はい、こんな感じ

```
--  
28 <!--** 以下Firebase **-->  
29   <script src="https://www.gstatic.com/firebasejs/5.7.0.firebaseio.js"></script>  
30   <script>  
31     // Initialize Firebase  
32     var config = {  
33       apiKey: "AIzaSyAK1Y5kbj1F1eIXhTpKFEW4rlGt16P0ahM",  
34       authDomain: "dev12-9bb66.firebaseio.com",  
35       databaseURL: "https://dev12-9bb66.firebaseio.com",  
36       projectId: "dev12-9bb66",  
37       storageBucket: "dev12-9bb66.appspot.com",  
38       messagingSenderId: "46587501915"  
39     };  
40     firebase.initializeApp(config);  
41   </script>
```

次のページへ

# ★ここ重要POINT！

## コピーしたコードの一部を削除

```
<script src="https://www.gstatic.com/firebasejs/5.9.1.firebaseio-app.js"></script>
```

「-app」の文字を削除

```
<script src="https://www.gstatic.com/firebasejs/5.9.1.firebaseio.js"></script>
```

<<解説>>

デフォルト 「firebase-app.js」 では以下のようにコアライブラリのみ読み込んでいて database.js などの処理は入っていない。そのため授業では、全部入り 「firebase.js」 を読み込んで使う。

```
<script src="https://www.gstatic.com/firebasejs/5.9.1.firebaseio-app.js"></script>
<!-- Add additional services that you want to use -->
<script src="https://www.gstatic.com/firebasejs/5.9.1.firebaseio-auth.js"></script>
<script src="https://www.gstatic.com/firebasejs/5.9.1.firebaseio-database.js"></script>
<script src="https://www.gstatic.com/firebasejs/5.9.1.firebaseio-firebase.js"></script>
<script src="https://www.gstatic.com/firebasejs/5.9.1.firebaseio-messaging.js"></script>
<script src="https://www.gstatic.com/firebasejs/5.9.1.firebaseio-functions.js"></script>
```

<https://firebase.google.com/docs/web/setup?hl=ja>

# Chat設定

# 11. コンソール画面

The screenshot shows the Firebase Project Overview page for a project named 'dev14'. The left sidebar contains navigation links for various services: Authentication, Database, Storage, Hosting, Functions, ML Kit, Crashlytics, Performance, Test Lab, Analytics, Dashboard, Events, Conversions, Predictions, and A/B Testing. The main content area displays the project summary for 'dev14' on a 'Spark プラン'. It shows that there is 1 個のアプリ (1 app) and a button to '+ アプリを追加' (Add app). A large text box says 'アプリに追加するプロダクトを選択します' (Select the product to add to the app). Below it, another text box says 'アプリデータを瞬時に保存して同期' (Save and sync app data instantly). There are cards for 'Authentication' (ユーザーの認証と管理) and 'Cloud F' (次世代の Re).

Firebase

Project Overview

開発

- Authentication
- Database
- Storage
- Hosting
- Functions
- ML Kit

品質

Crashlytics, Performance, Test Lab

アナリティクス

Dashboard, Events, Conversions, A...

拡大

- Predictions
- A/B Testing

dev14

dev14 Spark プラン

1 個のアプリ + アプリを追加

アプリに追加するプロダクトを選択します

アプリデータを瞬時に保存して同期

Authentication

ユーザーの認証と管理

Cloud F

次世代の Re

# 12. 「Authentication」→「ログイン方法」→「匿名」の順番に選択

The screenshot shows the Firebase console's Authentication interface. The left sidebar has a dark theme with sections for Project Overview, Development (selected), Database, Storage, Hosting, Functions, Stability, Analytics, and Growth. The main area is titled 'Authentication' and has tabs for Users, Sign-in method (which is selected and highlighted in blue), Templates, and Usage. The 'Sign-in method' tab displays a table of providers:

ログイン プロバイダ	ステータス
メール / パスワード	無効
電話番号	無効
Google	無効
Facebook	無効
Twitter	無効
GitHub	無効
匿名	無効

A large red arrow points from the 'Authentication' button in the sidebar to the 'Sign-in method' tab in the header. Another red arrow points from the 'Anonymous' row in the table to the 'Anonymous' row, which is highlighted with a red box.

# 13. 「有効にする」 → 「保存」 の順番でクリック

The screenshot shows the Firebase console's Authentication screen for a project named "mychat". On the left sidebar, under the DEVELOP section, the "Authentication" option is selected. The main content area displays several provider configurations: Google (disabled), Facebook (disabled), Twitter (disabled), GitHub (disabled), and a section for "匿名" (Anonymous) which is currently enabled. A red box highlights the "有効にする" (Enable) toggle switch for the Anonymous provider. Below this, a note explains that enabling anonymous authentication allows users to sign in without providing authentication information, while still enforcing security rules. At the bottom right, there are "キャンセル" (Cancel) and "保存" (Save) buttons, with the "保存" button also highlighted by a red box.

Firebase

mychat ▾ Authentication ドキュメントに移動

Project Overview |

DEVELOP

Authentication

Database

Storage

Hosting

Functions

STABILITY Crash Reporting, Performance, Test ...

ANALYTICS Dashboard, Events, Audiences, Attrib...

Authentication

Google 無効

Facebook 無効

Twitter 無効

GitHub 無効

匿名

有効にする

アプリケーションで匿名ゲスト アカウントを有効にします。これにより、認証情報の入力を要求すことなく、ユーザー固有のセキュリティ ルールおよび Firebase ルールを強制できます。 詳細

キャンセル 保存

# 14. 「Database」→「データベースの作成」

The screenshot shows the Firebase console interface for creating a new database.

- Left Sidebar:** Lists various Firebase services: Project Overview, Authentication, Database (highlighted with a red box), Storage, Hosting, Functions, ML Kit, Quality (Crashlytics, Performance, Test L...), Analytics (Dashboard, Events, Conversions,...), and Growth (Predictions, A/B Testing, Cloud Messaging, In-App Messaging).
- Top Bar:** Shows the project name "dev12" and the "Database" tab.
- Cloud Firestore Section:** Contains a note: "Cloud Firestore で実現できること 詳細".
- Realtime Database Section:** Contains a note: "Realtime Database". It says "Firebase 独自のデータベース Firestore と同様に、リアルタイム同期をサポートしています". It includes a "ドキュメントを表示" button and a "データベースを作成" button.
- Bottom Section:** Shows "デベロッパー向けのその他の機能" with icons for Predictions, A/B Testing, Cloud Messaging, and In-App Messaging.

A large red box highlights the "Database" service in the sidebar and the "Realtime Database" section in the main content area. A red arrow points from the "Database" sidebar item to the "Realtime Database" section.

# 15. 「ルール」 → 「テストモード」に設定

The screenshot shows the Firebase Realtime Database security rules configuration dialog. The dialog title is "Realtime Database のセキュリティ ルール". It contains two options:

- ロックモードで開始  
読み取りと書き込みをすべて拒否し、データベースを非公開で作成します
- テストモードで開始  
読み取りと書き込みをすべて許可し、設定をすばやく行います

A red box highlights the "テストモードで開始" option. A red arrow points from this box to the "有効にする" (Enable) button at the bottom right of the dialog. To the right of the dialog, a code snippet shows the generated security rules:

```
{  
  "rules": {  
    ".read": true,  
    ".write": true  
  }  
}
```

A yellow callout box with an exclamation mark provides a note: "データベース参照を所有しているユーザーなら誰でも、データベースの読み取りや書き込みを行えるようになります".

The background of the dialog shows the Cloud Firestore interface.

これで「匿名」の誰でもチャット参加することが可能になりました。  
次にチャットの最低限のコードを実装していきましょう。

# 16. DB作成完了

The screenshot shows the Firebase Realtime Database console for a project named "gsdemo". The "Database" tab is selected, and the "Realtime Database" sub-tab is active. The "Data" tab is currently selected. A prominent red warning message is displayed: "⚠ セキュリティ ルールが公開として定義されているため、誰でもデータベース内のデータを窃取、変更、削除できます。" (Warning: Security rules are defined publicly, so anyone can read, change, or delete data from the database). Below the warning, there is a URL field containing "https://gsdemo-15ec4.firebaseio.com/" and a navigation bar with a plus sign, minus sign, and three dots. On the left sidebar, under the "Development" section, the "Database" option is highlighted.

# 画面作成

# 16. エディターを開きHTMLを追記します。

☆Emmet: [ div>div\*3 ] を使いdivのブロックだけ作りましょう！

```
9   <!-- コンテンツ表示画面 -->
10
11  <div>
12    <div>名前:<input type="text" id="uname"> </div>
13    <div>
14      <textarea id="text" cols="30" rows="10"></textarea>
15      <button id="send">送信</button>
16    </div>
17    <div id="output"></div>
18  </div>
19
```



HTMLを追記後、HTMLをブラウザで表示します。

# Chat処理記述

# 17. jqueryを使用できるようしています。

```
29
30  <!-- JQuery -->
31  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
32  <!-- JQuery -->
33
34
```

今回はCDN（Webサーバー上に用意されてるライブラリ）から読み込んでいます。  
※各自でライブラリをダウンロードして使ってもOKです。

# 18. リアルタイム通信の記述をしましょう。

```
35 <script>
36   // Your web app's Firebase configuration
37   var firebaseConfig = {
38     apiKey: "AIzaSyCMwhUYGz1LvKc5u2Q5w6u-1E2wvwu9_n0",
39     authDomain: "gsdemo-15ec4.firebaseio.com",
40     databaseURL: "https://gsdemo-15ec4.firebaseio.com",
41     projectId: "gsdemo-15ec4",
42     storageBucket: "gsdemo-15ec4.appspot.com",
43     messagingSenderId: "920619037354",
44     appId: "1:920619037354:web:8c70672e9a37f98be416d3"
45   };
46   // Initialize Firebase
47   firebase.initializeApp(firebaseConfig);
48   const ref = firebase.database().ref();
49 
```

`const ref = firebase.database().ref();`

の`ref();`を追加することでリアルタイム通信をおこないます。

# 19.送信ボタンのクリックイベントを作成

## ◇使うElement(要素)

- ・ ボタン #send
- ・ テキストボックス #uname
- ・ テキストエリア #text
- ・ div (メッセージ表示領域) #output



## ◇ボタンclickイベントを作成

```
57 //送信
58 $("#send").on("click",function(){
59     const uname = $("#uname").val();
60     const text = $("#text").val();
61     alert(uname+text); //取得確認
62 }
63 );
```

# 20.データ送信：処理を記述します。

```
57 //送信
58 $("#send").on("click",function(){
59     const uname = $("#uname").val();
60     const text = $("#text").val();
61     const msg = {
62         uname: uname,
63         text: text
64     };
65     ref.push(msg);
66 });

});
```

以下処理の流れ

- 58: id="send"をクリックしたら
- 59: id="uname" から入力データを取得
- 60: id="text"から入力データを取得
- 61~64: オブジェクト変数を作成
- 65: ref.push(オブジェクト変数); でデータ送信

# 23.データ受信：処理を記述します。

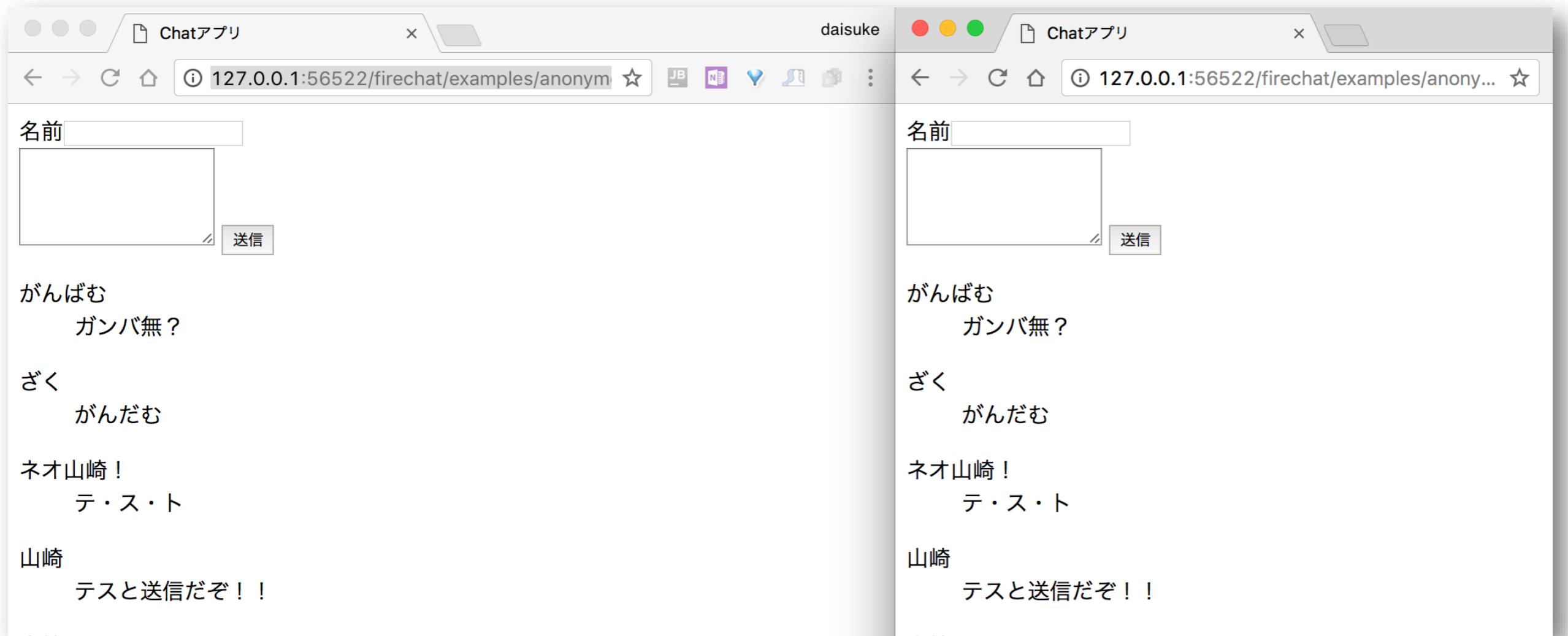
例) 受信完成コード

ref.on(); を使い送信してくるデータを監視

```
67
68 // "child_added:毎回1個", "value:毎回全てのデータを取得"
69 ref.on("child_added",function(data){ //引数でオブジェクトデータを受け取る
70     const v = data.val();           // .val()を使いオブジェクト変数を取得
71     const k = data.key;            // UniqueKEY : データベース参照
72     const h = '<p>' + v.uname + '<br>' + v.text + '</p>'; //html文字作成
73     $("#output").prepend(h);      // #outputに差し込む
74 });
75
```

# 24. チャット動作確認

ブラウザ2つで開き、チャット送信ができるか確認しましょう。



# 25. 管理画面→「Database」を表示

データ構造を見ることが出来ます。

The screenshot shows the Firebase Realtime Database interface. On the left, a sidebar lists various services: Overview, Analytics, Database (which is selected and highlighted with a red box and circled 1), Storage, Hosting, Functions, Test Lab, Crash Reporting, and Performance. The main area is titled "Realtime Database" and shows the database structure for the project "chatapp-1aa3a". A URL "https://chatapp-1aa3a.firebaseio.com/" is displayed above the tree view. The database structure is as follows:

- KitDXv-QN26PLZITReo
- KitDn9IEVX3TNDmYqG1
- KkNqu-RS8cEuP6YSR7F
- KkNr9YkYPifW\_gIVeV1
  - text: "ガンバ無?\n"
  - username: "がんばむ"

A large red arrow points from the "Database" button in the sidebar to the database root node "chatapp-1aa3a".

# EnterKeyで送信

# Enter Keyで送信する方法

keydownイベント：キー入力を取得

```
48 ... $("#text").on("keydown", function(e){  
49     ...     console.log(e);  
50     ...});
```

console画面で確認

```
▼ m.Event {originalEvent: KeyboardEvent, type: "keydown", timeStamp: 6131.685000000001,  
  altKey: false  
  bubbles: true  
  cancelable: true  
  char: undefined  
  charCode: 0  
  ctrlKey: false  
  currentTarget: textarea#text  
  data: undefined  
  delegateTarget: textarea#text  
  eventPhase: 2  
  handleObj: {type: "keydown", origType: "keydown", data: undefined, guid: 4, handler  
  isDefaultPrev  
  jQuery1113088  
  key: "Enter"  
  keyCode: 229  
  metaKey: false  
  originalEvent: KeyboardEvent {isTrusted: true, key: "Enter", code: "F13", location: "Left", repeat: false, nativeEvent: Object, preventDefault: [Function], stopPropagation: [Function]}  
  relatedTarget: undefined  
  shiftKey: false  
  target: textarea#text}
```

keyCode:13がEnter

e がキモ！

Enterを押したら  
送信を作つて見よう！

# そしてconsole.logがキモ

EnterKey の番号を取得したり、  
何処をクリックしたのか？など、X,Y座標も  
取得したり幅広く使います。

console.logを活用してデータを可視化する  
ことで開発スピードが上がります。

# ChatにIcon

初~中級向け

理解できていて次に進める人は以下ファイル参照  
サンプルファイル内 「firebase\_add.pdf」

# 課題発表

# Line風アプリ制作

## ●課題

---

◆ Line風アプリの課題最低限機能

1. 「名前 + 日時 + メッセージ」

※appendを使います（限定）

2. 「メッセージ表示領域を超えた処理」

※表示エリア : height:300px;overflow:auto; などでスクロール表示

3. 見た目の装飾

さらにあれば良いと思わる機能

「アイコン」、「翻訳」、「Map連携」、「Canvas連携」、「ジャンケン」とか？