



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

FireBase

課題実習タイム (Chatアプリ or リアルタイム通信)

❖ チュータリングタイム

オブジェクト

- Object -

JavaScriptのオブジェクト

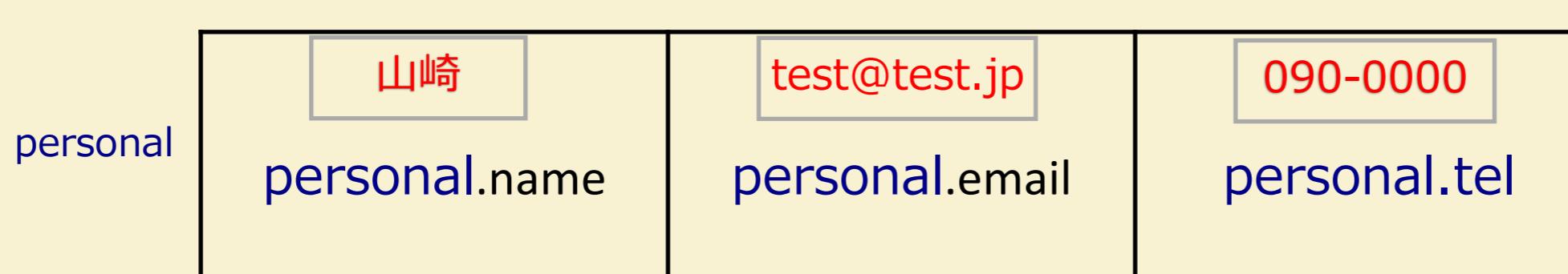
【 オブジェクト (Object) 】

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/>



The banner features a blue background with a stylized illustration of two people. On the left, a person in a yellow hoodie sits cross-legged, looking at a laptop. On the right, a person in a dark hoodie stands, pointing towards the center where several floating blue squares represent data or code. The overall theme is mobile and web app development.

Firebase で
モバイルとウェブ
アプリを成功させる

使ってみる

動画を見る

検索

Language ▾

コンソールへ移動

1.新規プロジェクト作成

The screenshot shows the Firebase console's main dashboard. At the top left is the Firebase logo. On the right, there is a link labeled "ドキュメントに移動". The central area features a section titled "最近のプロジェクト" (Recent Projects) with a large red box highlighting the first item. This item contains a blue plus sign icon and the text "プロジェクトを追加" (Add project). Below this is a link "デモ プロジェクトを見る" (View demo project). To the right of this are two other project cards, each with a double slash icon. Below this section is another row of three project cards, each with a single slash icon. At the bottom of the page, there is a section titled "すべての Firebase プロジェクト" (All Firebase projects).

2.新規プロジェクト作成

× プロジェクトの作成（手順 1/3）

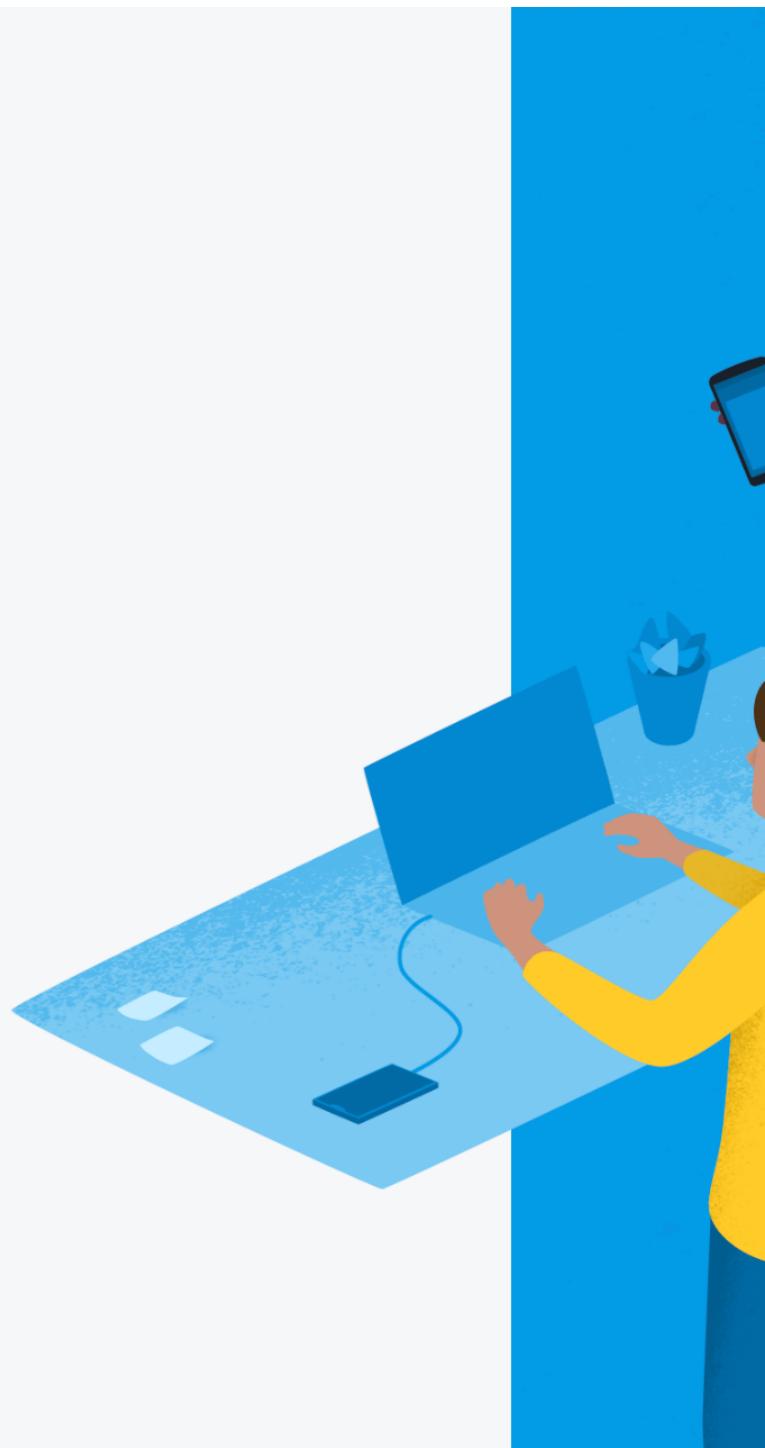
まずプロジェクトに名前を付
けましょう

プロジェクト名

labs-f2878

↓

続行



3.新規プロジェクト作成

× プロジェクトの作成（手順 2/3）

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 [?](#)
-  クラッシュに遭遇していないユーザー [?](#)
-  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 [?](#)
-  イベントベースの Cloud Functions トリガー [?](#)
-  ユーザー行動の予測 [?](#)
-  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
推奨

前へ

続行

4.新規プロジェクト作成

× プロジェクトの作成（手順 3/3）

Google アナリティクスの構成

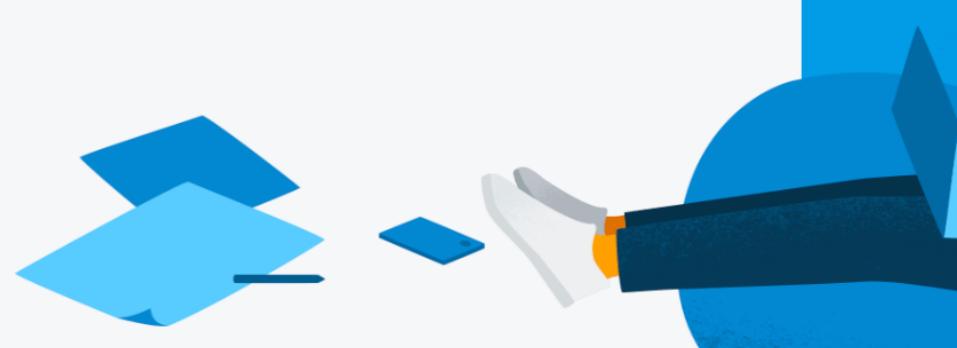
Google アナリティクス アカウントを選択または作成します [?](#)

プロジェクトを作成すると、選択した Google アナリティクス アカウントに新しい 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

4.新規プロジェクト作成



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



Firebaseに接続する大事にKEYになります。



7. コピーボタンでCODEをコピーしておく。

- × ウェブアプリに Firebase を追加

- ✓ アプリの登録
- 2 Firebase SDK の追加

以下コードをコピー

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ます。この作業は Firebase サービスを使用する前に行ってください。

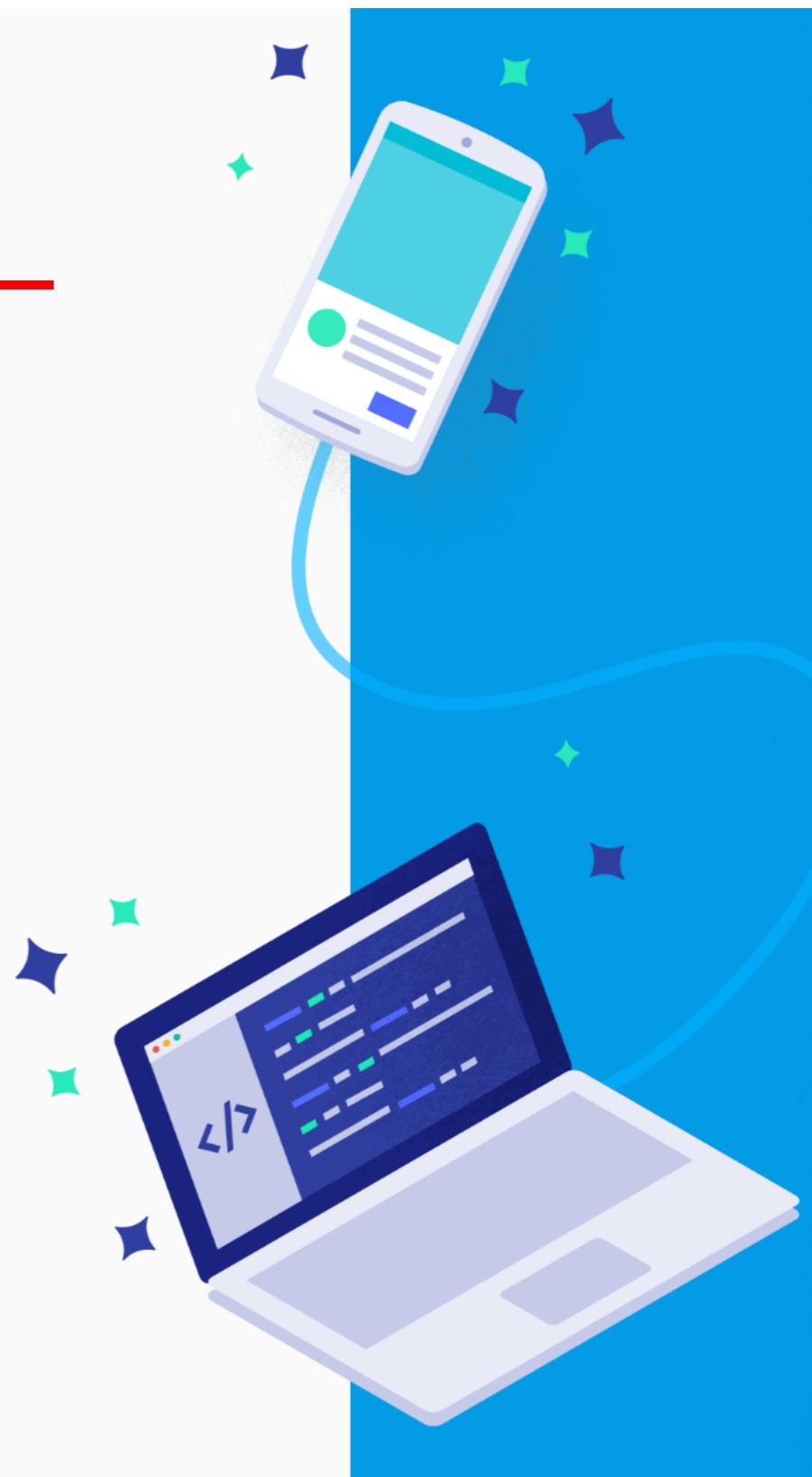
```
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/6.0.2.firebaseio.js"></script>

<!-- 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](#)

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

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

10. コンソールに移動（次の画面へ）

- × ウェブアプリに Firebase を追加

✓ 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ます。この作業は Firebase サービスを使用する前に行ってください。

```
<!-- 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.firebaseioapp.com",
    databaseURL: "https://chats-e9be4.firebaseioio.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](#)

コンソールに進む



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 page for a project named "mychat". The left sidebar lists "Project Overview", "DEVELOP" (with "Authentication" selected), "STABILITY", "ANALYTICS", "GROW", and "Spark". The main "Authentication" tab is active. The "Sign-in methods" section lists several providers: Email / Password (disabled), Phone number (disabled), Google (disabled), Facebook (disabled), Twitter (disabled), GitHub (disabled), and Anonymous (disabled). A red arrow points from the "Authentication" tab in the sidebar to the "Sign-in methods" tab in the main content area. Another red arrow points from the "Anonymous" provider row to the "Anonymous" tab at the bottom of the list.

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

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 about实现 (Achievement) and a "Cloud Firestore" logo.
- Main Content Area:**
 - A large callout box highlights the choice between "Cloud Firestore" and "Realtime Database".
 - The "Realtime Database" section is also highlighted with a red box, showing its description: "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" (Firebase's own database. Supports real-time synchronization like Firestore). It includes a "ドキュメントを表示" (View document) link and a "データベースを作成" (Create database) button.
 - Below the main sections, there is a heading "デベロッパー向けのその他の機能" (Developer-oriented other features) followed by three small icons representing Predictions, A/B Testing, and Cloud Messaging.

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. 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" (If you own a database reference, anyone can read or write to it). The background shows the Firebase console interface with the "Database" tab selected.

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。

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  <div>
11    ... <div>名前<input ·type="text" ·id="username"></div>
12  <div>
13    ... <textarea ·rows="5" ·id="text"></textarea>
14    ... <button ·id="send">送信</button>
15  </div>
16  ... <!--・リスト表示・-->
17  ... <div ·id="output"></div>
18 </div>
19 <!--/・コンテンツ表示画面・-->
```

A screenshot of a web browser window displaying a simple HTML form. The form consists of a single text input field with the placeholder text "名前" (Name) and a button labeled "送信" (Send). The browser interface shows the code structure above the rendered form.

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。

```
21
22  <!-- 以下JavaScript領域 -->
23  <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
24  <script src="https://www.gstatic.com/firebasejs/3.9.0.firebaseio.js"></script>
25  <script>....|
26  //Firebase接続
27 ▼ var config = {
28    apiKey: "AIzaSyCy_qULSiL1p26Xg-NETlmRPLy5zJtuGjU",
29    authDomain: "chatapp-1aa3a.firebaseio.com",
30    databaseURL: "https://chatapp-1aa3a.firebaseio.com",
31    projectId: "chatapp-1aa3a",
32    storageBucket: "chatapp-1aa3a.appspot.com",
33    messagingSenderId: "260581366983"
34  };
35  firebase.initializeApp(config);
36
```

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。

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

※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。

```
25 <script>....  
26 //Firebase接続  
27 ▼ var config = {  
28   .... apiKey: "AIzaSyCy_qULSiL1p26Xg-NETlmRPLy5zJtuGjU",  
29   .... authDomain: "chatapp-1aa3a.firebaseio.com",  
30   .... databaseURL: "https://chatapp-1aa3a.firebaseio.com",  
31   .... projectId: "chatapp-1aa3a",  
32   .... storageBucket: "chatapp-1aa3a.appspot.com",  
33   .... messagingSenderId: "260581366983"  
34 };  
35 firebase.initializeApp(config);  
36  
37 //MSG送受信準備  
38 var newPostRef = firebase.database().ref();
```

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

のref();を追加することでリアルタイムに通信するようになります。

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 | #username |
| • 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

```
40 //Submit:MSG送信
41 ▼ $("#send").on("click", function() {
42
43     ...
44
45     ...
46 });
47
```

20.データ送信：処理を記述します。

記述式)

```
送信オブジェクト.push({
    送信プロパティ名: 値,
    送信プロパティ名: 値
});
```

pushオブジェクトを使用して送信。

```
40 //Submit:MSG送信
41 ▼ $("#send").on("click", function(){
42 ▼     newPostRef.push({
43         .....username: $("#username").val(),
44         .....text: $("#text").val()
45     });
46     .....$("#text").val("");
```

21.データ受信：処理を記述します。

onメソッドの 'child_added' イベントを取得し、受信処理を行います。
※ 'child_added' はFirebaseが用意しているイベントです。

```
56 //4. MSGデータ受信
57 //child_added:毎回1個//value:毎回全てのデータを取得
58 newPostRef.on('child_added',function(data) {
59     let v = data.val();    //データ取得
60     let k = data.key;      //ユニークKEY取得(今回は使わない)
61
62
63
64
65
66});
```

22. データ受信：処理を記述します。

データ取得方法)

以下のように「`data.val()`」で受信し、`v`変数に代入する。

```
var v = data.val(); //データ取得  
var k = data.key; //ユニーク KEY 取得
```



`v.username`

`v.text`

「`v.送信プロパティ名`」で取得します。

※ユニーク KEY は Firebase 側で自動で割り振られる KEY です。

送信データには全て KEY が割り振られます。

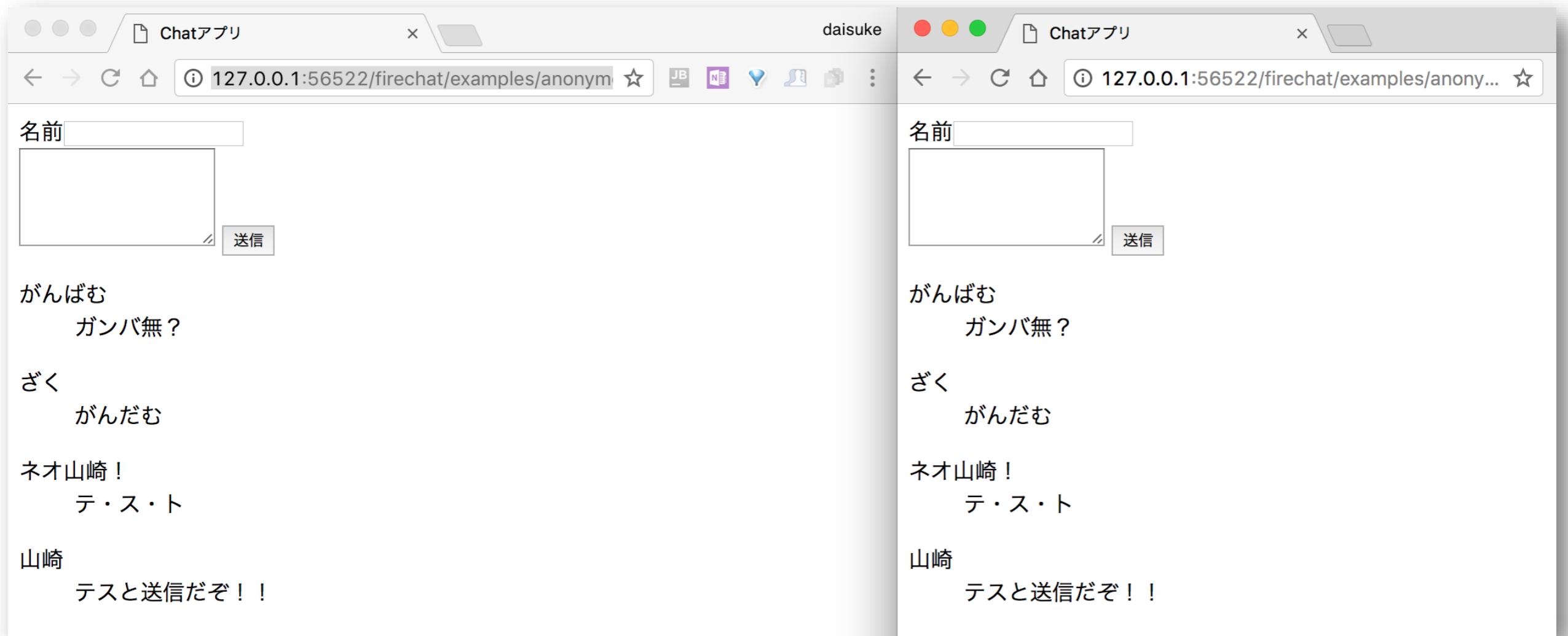
23.データ受信：処理を記述します。

例) 受信完成コード

```
56 //4. MSGデータ受信
57 //child_added:毎回1個//value:毎回全てのデータを取得
58 newPostRef.on('child_added',function(data) {
59     let v = data.val();    //データ取得
60     let k = data.key;      //ユニークKEY取得(今回は使わない)
61
62     //Timelineメッセージ作成
63     let str = '<p>' + v.username + '<br>' + v.text + '</p>';
64     $('#output').prepend(str);
65
66});
```

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, type: "keydown"}, 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？