

Deploy App Flutter Android para Flutter Web no Firebase Hosting

Este é um artigo em constante atualização e a sua contribuição fará muita diferença. **Então deixe seus comentários abaixo.** Para que ele fique cada dia melhor e atualizado. E deixe seu clap, se te ajudou, pois pode ajudar outros.

O problema

O problema que vamos, ou tentaremos, clarear a solução aqui é, de que, quando vamos desenvolver um app a primeira definição é a plataforma. Se for apenas Flutter Android, seguiremos as orientações para lançamento em PlayStore. *E NÃO trataremos disto neste artigo.* Mas se vamos desenvolver na mesma base de código um app em Flutter para Android e também para Web aí precisamos de certas configurações para execução no Android e deploy na Web no Firebase Hosting. Não trataremos aqui de código interno do app em Dart do tipo `if(kIsWeb){}else if(TargetPlatform.android){}else{throw UnsupportedError()}`. Vamos focar apenas em configurações externas para execução no Android e deploy Web no Firebase Hosting com a mesma base de código para recursos e packages que devem funcionar nestas duas plataformas. Os recursos/packages ate agora contemplados nas configurações sugeridas a seguir são: loginGoogle e loginEmailSenha. A medida que a publicação for recebendo contribuições vou aumentando os recursos/packages e as configurações para execução no Flutter Android e deploy Flutter Web no Firebase Hosting para aceitar recursos tais como Firestore, Storage, etc.

O App

Se um app é igual ao counter ele deve funcionar praticamente em qualquer plataforma que o Flutter atende hoje. Mas se anexarmos recursos/packages ele se limita a algumas plataformas apenas. E é neste ponto que se os recursos/packages de nosso app teste forem iguais ao seu; o seu app deverá funcionar com estas configurações aqui apresentadas para execução no Flutter Android e deploy Flutter Web no Firebase Hosting.

Vamos considerar como base de nossa conversa um simples app **ToDo**. Daqueles em que você cadastra uma tarefa, nosso modelo básico terá (id_uuid, description_string, date_datetime, done_bool) e depois gerencia a edição destas tarefas. Sendo a interface minimalista. Recursos do App: 1) login via email e senha no Firebase Authentication; 2) login com google no Firebase Authentication; 3) armazenamento local com Hive; 4) outras que iremos acrescentando e atualizando neste artigo. O foco de ter estes recursos/packages é proporcionar um exemplo simples para execução do app no Flutter Android e Flutter Web no Firebase Hosting.

Este artigo foi inspirado numa excelente aula do Prof [Rodrigo Rahman](#) na Academia do Flutter, ou em <https://www.youtube.com/channel/UC5hvPObwya8kzWWB-wmVIXg>. Vale a pena conhecer o curso e a comunidade. Alguns ajustes no código para GetX estão sendo feitos com base nos artigos do Prof [Marcus Brasizza](#) (Brasizza#7615), aqui no medium. E o proximo passo e fazer este ToDo baseado no fantástico artigo Prof Felipe Nanclarez (filipe.nanclarez#1941) em <https://dev.to/filipenanclarez/eu-tenho-um-sonho-offline-first-em-flutter-4nf8>. Meu discord é catalunha#5282. Q coisa estou a disposição.

O básico

O começo é básico, mas depois vamos precisar de tudo isto quando executarmos a mesma base de código em Flutter Android e Flutter Web.

Para criação de nosso projeto usaremos o terminal com o comando:

```
1 $ flutter create --project-name=todo_medium --org br.c  
2 // --org -> Nome do pacote android ou caminho inverso  
3 // todo_medium -> nome do app que geralmente é o mesmo  
4 // --platforms -> projeto criado apenas para android e  
5 // -- -- a linguagem para app android será kotlin, e a
```

A estrutura de pastas e arquivos se parecerá neste momento com esta. Mas iremos alterar algumas coisas ao longo da configuração.

```
1 [+] .  
2 [+] ..  
3 [+] .dart_tool  
4 [+] .idea  
5 [+] android  
6 [+] lib  
7 [+] test  
8 [+] web  
9 .gitignore  
10 .metadata  
11 .packages
```

Código: folders01

Agora vamos criar nosso projeto no firebase, neste link <https://console.firebase.google.com/u/0/>.



Ilustração: 01

Informaremos o nome do projeto. E as demais configurações siga as ilustrações ou altere de acordo com sua realidade. Qualquer dúvida avisa pra gente.



Ilustração: 02



Ilustração: 03

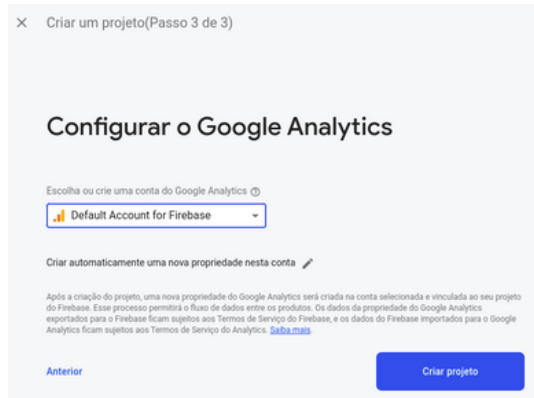


Ilustração: 03

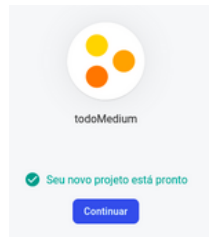


Ilustração: 04

O app Flutter para Android

O sistema retorna a página inicial e adicionaremos um App Android. Clique então no ícone do Android.

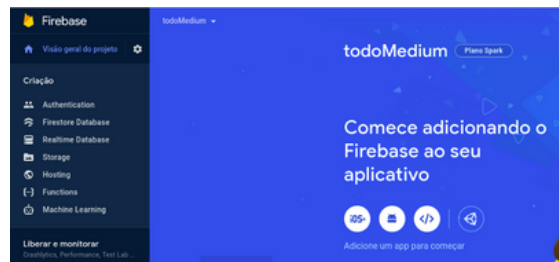


Ilustração: 05

Etapa 1. Temos o registro do App. É importante pois sem ele não conseguimos fazer login com google e ter outras funções.

Ilustração: 06

Para os três campos fiz um resumo no código a seguir. Em destaque no campo03 vamos precisar deste link <https://developers.google.com/android/guides/client-auth>.

Campo 01:

Acesse ao projeto neste caminho e leia do arquivo a informação necessária.

```
[*] android/app/build.gradle
    android {
        defaultConfig {
            applicationId "br.com.empresatodo_med"
```

Campo 02:

Um valor qualquer.

Campo 03:

Acesse este link

<https://developers.google.com/android/guides/client-auth> e em 'Using Keytool on the certificate' na parte Linux veja o comando a seguir. E logo abaixo sua saída. Lembre-se que a senha é **android**

```
$ keytool -list -v -alias androiddebugkey -keystore
```

Código: TelaRegistroApp

Etapa 2. Faça o download do arquivo de configurações para seu projeto Flutter Android. Muitas coisas nele usaremos também no projeto Flutter Web. O arquivo de configurações deve ser colocado neste local: **android/app/google-services.json**. Reveja as pastas do projeto.



Ilustração: 07

Etapa 3. Nestas telas iremos fazer muitas alterações em arquivos importantes então fique atento. Veja meu arquivo código para instruções logo em seguida.



Ilustração: 08

Lembra que nós escolhemos como linguagem para android o kotlin, opção -a ao criar o projeto no Flutter. Então não esqueça de marcá-la.



Ilustração: 09

Siga as alterações oficiais conforme edições neste arquivos logo a seguir. Uma alteração complementar também será feita neste

momento. Para evitar erros de minSdkVersion, conforme explanado pelo Rodrigo Rahman no video <https://youtu.be/EGTKnkiEp4>. Veja meus comentários. Qualquer dúvida avisa pra gente.

Dentro da pasta android tem muitos arquivos importantes com mesmo nome mas em pastas diferentes.

Vamos alterar dois deles a saber:

- android/build.gradle
- android/app/build.gradle

Então vamos com muita calma nesta hora. Vamos alterar primeiro o

android/build.gradle

Não altere as demais linhas. Acrescente apenas o informado no comentado.

```
buildscript {  
    repositories {  
        google() // Verifique se tem esta declaração.  
        ...  
    }  
    dependencies {  
        ...  
        classpath 'com.google.gms:google-services:4.3'  
    }  
}  
allprojects {  
    ...  
    repositories {  
        google() // Verifique se tem esta declaração.  
        ...  
    }  
}
```

e agora o segundo arquivo. Faça as alterações oficiais da configuração.

android/app/build.gradle

```
apply plugin: 'com.android.application' // locali  
...  
apply plugin: 'com.google.gms.google-services' //  
  
dependencies {  
    ...  
    // Import the Firebase BoM  
    implementation platform('com.google.firebase:fi  
    // Add the dependency for the Firebase SDK for  
    // When using the BoM, don't specify versions i  
    implementation 'com.google.firebase:firebase-an  
    // Add the dependencies for any other desired F  
    // https://firebase.google.com/docs/android/set  
}
```

Codigo: TelaAdicionarSdk

Na proxima tela apenas clique em continuar no console.

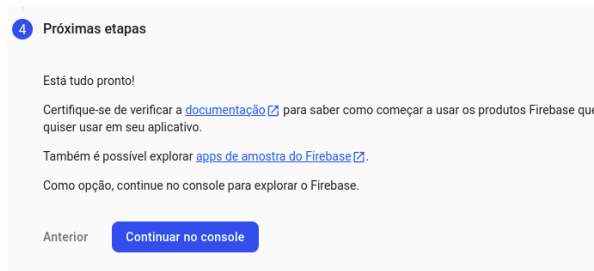


Ilustração: 10

Precisamos agora configurar as contas do Firebase Auth para fins de autenticação via email/senha e google. Clique em primeiros passos.

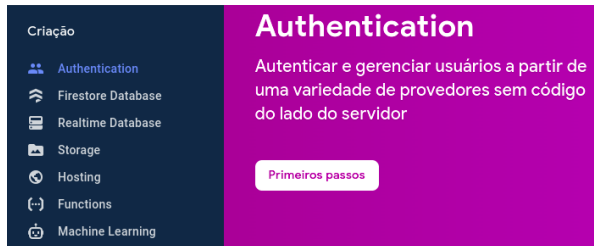


Ilustração: 11

Escolha Email/senha na próxima tela:

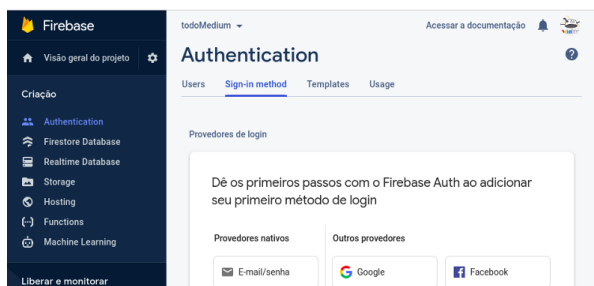


Ilustração: 12

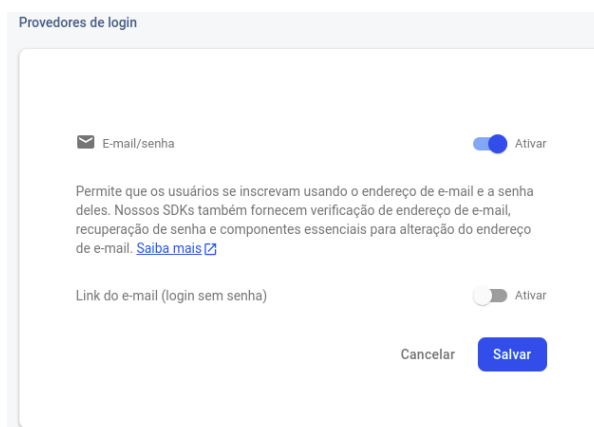


Ilustração: 13

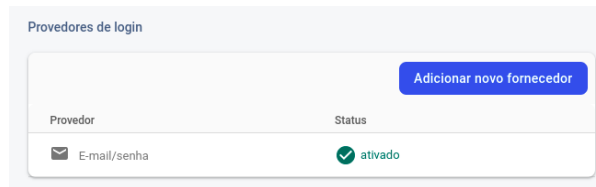


Ilustração: 14

Clique novamente em adicionar novo provededor de acesso e escolha Google. Não esqueça de escolher o email.

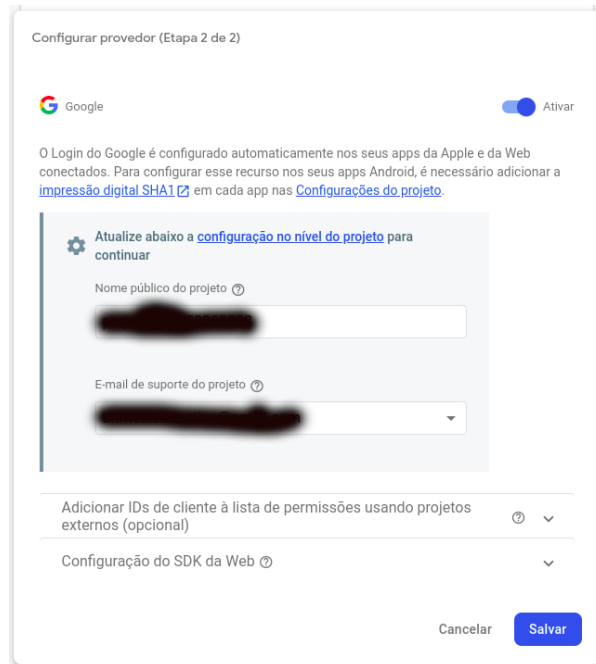


Ilustração: 15

Ao final a configuração deverá ficar assim:

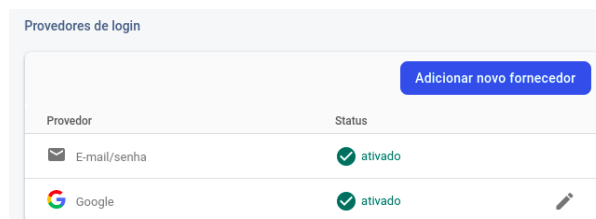


Ilustração: 16

Terminamos as configurações no Firebase. Para os recursos listados ate agora.

Voltemos ao projeto Flutter e vamos atualizar o pubspec.yaml com estes packages do codigo pubspec. Não precisaríamos de tudo isto mas para que o simples, **ToDo**, fique bonito e funcional precisaremos destes packages ai.

Futuramente iremos colocar outros pois pretendo que este artigo envolva outros recursos do Firebase num app Flutter Android e Flutter Web. Na mesma base de código e com configurações para Execução em Android e deploy na Web.


```
1
2 dependencies:
3   flutter:
4     sdk: flutter
5   # Localizations
6   flutter_localizations:
7     sdk: flutter
8   # Gerenciador de estados
9   get: ^4.6.1
10  # LocalStorage
11  hive: ^2.0.6
12  hive_flutter: ^1.1.0
13  # remoteStorage
14  firebase_auth: ^3.3.9
15  firebase_core: ^1.13.1
16  # Google
17  google_sign_in: ^5.2.4
18  google_fonts: ^2.3.1
19  # outros
```

Codigo: pubspec

O código fonte está disponível neste link, ??clique aqui??, não vamos perder tempo falando sobre ele, não é foco deste artigo, e ele deve apenas usar os recursos/packages listados. E deve ser acrescentado na pasta, lib/app e não esqueça os assets. Nosso foco é executar o App no Android e na Web com login via email/senha e google, lembra ?

Agora vamos alterar e criar outros arquivos.

```
[+] .
[+] ..
[+] .dart_tool
[+] .idea
[+] android
[+] assets // copie os assets nesta pasta.
[+] lib
    [+] app // copie o código fonte nesta pasta.
    main.dart // altere este arquivo como mostra
    firebase_options.dart // acrescente este arq
[+] test
[+] web
.gitignore
.metadata
```

Codigo: folders02

Primeiro o main.dart. Como a seguir:

```
1  import 'package:firebase_core/firebase_core.dart';
2  import 'package:flutter/material.dart';
3  import 'package:get/get.dart';
4  import 'package:todo_getx_hive/app/routes.dart';
5  import 'package:todo_getx_hive/app/views/core/ui/theme.dart';
6  import 'package:flutter_localizations/flutter_localizations.dart';
7  import 'firebase_options.dart';
8
9  void main() async {
10    WidgetsFlutterBinding.ensureInitialized();
11    await Firebase.initializeApp(
12      options: DefaultFirebaseOptions.currentPlatform,
13    );
14    runApp(MyApp());
15  }
16
17  class MyApp extends StatelessWidget {
18    const MyApp({Key? key}) : super(key: key);
19
20    @override
21    Widget build(BuildContext context) {
22      return GetMaterialApp(
23        title: 'Flutter Demo',
24        theme: ThemeConfig.theme(),
```

Codigo: main

E depois o firebase_options.dart. Como a seguir. Ocultei alguns valores mas vou mostrar a seguir onde obté-los.

```

1  import 'package:firebase_core/firebase_core.dart' show
2  import 'package:flutter/foundation.dart'
3      show defaultTargetPlatform, kIsWeb, TargetPlatform
4
5  /// Default [FirebaseOptions] for use with your Firebase
6  ///
7  /// Example:
8  /// ```dart
9  /// import 'firebase_options.dart';
10 /// // ...
11 /// await Firebase.initializeApp(
12 ///   options: DefaultFirebaseOptions.currentPlatform
13 /// );
14 /// ```
15 class DefaultFirebaseOptions {
16   static FirebaseOptions get currentPlatform {
17     if (kIsWeb) {
18       return web;
19     }
20     // ignore: missing_enum_constant_in_switch
21     switch (defaultTargetPlatform) {
22       case TargetPlatform.android:
23         return android;
24       case TargetPlatform.iOS:
25         throw UnsupportedError(
26           'DefaultFirebaseOptions have not been configured for iOS,
27           'you can reconfigure this by running the Flutterfire CLI
28         );
29       case TargetPlatform.macOS:
30         throw UnsupportedError(
31           'DefaultFirebaseOptions have not been configured for macOS,
32           'you can reconfigure this by running the Flutterfire CLI
33         );
34     }
35
36     throw UnsupportedError(
37       'DefaultFirebaseOptions have not been configured for this platform,
38       'you can reconfigure this by running the Flutterfire CLI

```

Codigo: firebase_options

De volta ao Firebase clique na engrenagem e escolha configurações do projeto.

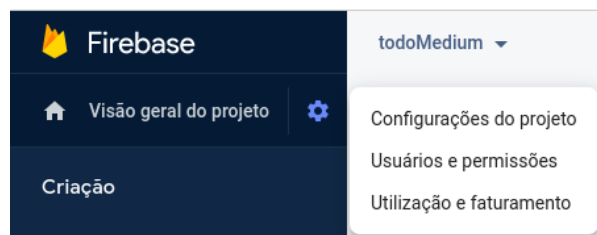


Ilustração: 17

Você verá esta página.

Seu projeto

Nome do projeto: todoMedium

Código do projeto: **projectId**

Número do projeto: [redacted]

Local padrão dos recursos do GCP: Ainda não selecionado

Chave de API da Web: **apiKey do Android**

Ambiente

Esta configuração personaliza o projeto para diferentes fases do ciclo de vida do aplicativo

Tipo de ambiente: Não especificado

Configurações públicas

Essas configurações controlam instâncias do seu projeto que são mostradas ao público

Nome exibido ao público: project: **messagingSenderId**

E-mail para suporte: [redacted]

Ilustração: 18

Aqui já encontramos o `projectId`, `apiKey` para android, `messagingSenderId`. Vamos então a tela logo abaixo da anterior.

Seus aplicativos

Adicionar aplicativo

Apps Android

ToDoMedium
br.com.empresa.todo_me...

Configuração do SDK

Precisa reconfigurar os SDKs do Firebase para seu app? Consulte novamente as instruções de configuração do SDK ou apenas faça o download do arquivo de configuração que contém as chaves e identificadores do seu app.

Ver instruções do SDK

google-services.json

ID do aplicativo: **appId**

Apelido do app: ToDoMedium

Nome do pacote: br.com.empresa.todo_medium

Impressões digitais do certificado SHA: [redacted] Tipo: SHA-1

Adicionar impressão digital

Remover este aplicativo

Ilustração: 19

Aqui temos o `appId`.

Você lembra daquele arquivo de configurações do Firebase que você salvou em `android/app/google-services.json`, pois é, ele também te fornece os valores que precisamos no `firebase_options.dart` na seguinte relação: `apiKey` para Android na variável `current_key`; `appId` na variável `mobilesdk_app_id`; `messagingSenderId` na variável `project_number`; `projectId` na variável `project_id`; `storageBucket` na variável `storage_bucket`.

O app Flutter para Web

Faltou então o apiKey para Web. Já que nosso foco é criar um app em Flutter Android e exportar para Web. Vamos então criar o App Flutter Web para este mesmo código. Voltemos então para o Firebase e clique em adicionar aplicativo conforme abaixo e escolha Web, ou </>.



Ilustração: 20

Você será redirecionado para esta página e marque conforme as imagens.

×

Adicionar o Firebase ao seu app da Web

1

Registrar app


Apelido do app ⓘ

ToDoMedium

☒

Além disso, configure o **Firebase Hosting** para este app. [Saiba mais](#) ⓘ

É possível configurar o recurso depois. Faça isso a qualquer momento sem custos financeiros.



todomedium-a05b4 (Nenhuma implantação i

▼

Registrar app

Ilustração: 21

Neste ponto eu me esqueci de colocar um nome melhor para app da web. Então ficou este nome feio todomedium-a05b4. Mas você pode escolher um melhor.

Observe agora que teremos a apiKey é para a versão Web e deve ir para nosso arquivo **firebase_options.dart** completando a ultima informação que faltava. As demais são iguais a Android



Ilustração: 22

Mas veja este arquivo manda você copiar este conteúdo para o arquivo **index.html** e vamos fazer isto. E observe que as linhas 2, 3 e 10 foram acrescentadas também, coisa da experiência...

```
1 ...
2 <script src="https://apis.google.com/js/platform.js" async defer></script>
3 <meta name="google-signin-client_id" content="...">
4 </head>
5 <body>
6 <script type="module">
7   // Import the functions you need from the SDKs you need
8   import { initializeApp } from "https://www.gstatic.com/firebasejs/9.6.8/firebase-app.js";
9   import { getAnalytics } from "https://www.gstatic.com/firebasejs/9.6.8/firebase-analytics.js";
10  import { } from "https://www.gstatic.com/firebasejs/9.6.8/firebase-database.js";
11
12  // TODO: Add SDKs for Firebase products that you want to use
13  // https://firebase.google.com/docs/web/setup#available-libraries
14
15  // Your web app's Firebase configuration
16  // For Firebase JS SDK v7.20.0 and later, measurementId is optional
17  const firebaseConfig = {
18    apiKey: "...",
19    authDomain: "...",
20    projectId: "...",
21    storageBucket: "...",
```

Código: index

Brincadeira.

A linha 2: ainda não me lembro. Mas estou resolvendo isto. :-)

A linha 3 é obtida no arquivo **android/app/google-services.json** na variável { **client_id**: "...", **client_type**: 3}. Pois tem dois client_id e pegue este do type=3. O valor de ... também pode ser obtido em outra fonte. Mostrarei depois.

A linha 10 foi obtida nestes links fornecidos no rodapé da ilustração anterior. O primeiro aponta para <https://firebase.google.com/docs/web/setup?authuser=0&hl=pt> e já resolve.

Você está usando o npm e um bundler como Webpack ou Rollup? Consulte o [SDK modular](#).

Saiba mais sobre o Firebase para Web: [Primeiros passos](#), [Referência da API Web SDK](#), [Amostras](#)

Ilustração: 23

Continuemos. Execute então o comando no terminal do projeto: **npm install -g firebase-tools** conforme orientado a seguir. Não se esqueça de colocar sudo pois estaremos atualizando pacotes a nível de root.

3

Instalar a CLI do Firebase

Para hospedar seu site com o Firebase Hosting, é necessário usar a Firebase CLI (uma ferramenta de linha de comando).

Execute o seguinte comando do [npm](#) para instalar a CLI ou atualizar para a versão mais recente da CLI.

```
$ npm install -g firebase-tools
```

Não está funcionando? Consulte a [referência da Firebase CLI](#) ou altere suas [permissões de npm](#)

Próxima

Ilustração: 24

Na etapa 4 basta executar os comandos na sequência orientada. Mas vamos falar sobre cada um deles.

4

Implantar no Firebase Hosting

É possível implantar agora ou [depois](#). Para implantar agora, abra uma janela de terminal e navegue ou crie um diretório raiz a partir do seu app da Web.

Faça login no Google

```
$ firebase login
```

Iniciar seu projeto

Execute este comando no diretório raiz do seu app:

```
$ firebase init
```

Quando tudo estiver pronto, implante seu app da Web

Inclua seus arquivos estáticos (por exemplo, HTML, CSS, JS) no diretório de implantação do app (o padrão é "público"). Depois, execute este comando a partir do diretório raiz do app:

```
$ firebase deploy
```

Após a implantação, veja seu app em [todomedium-a05b4.web.app](#).

Precisa de ajuda? Confira a [documentação do Hosting](#)

Continuar no console

Ilustração: 25

Ao executar **firebase login** ele irá apenas logar em sua conta do Firebase.

Ao executar **firebase init** teremos estas opções. Escolha Hosting sem GitHub Actions.

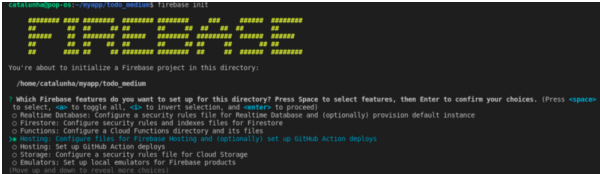


Ilustração: 26

Escolha projeto existente e selecione seu projeto.

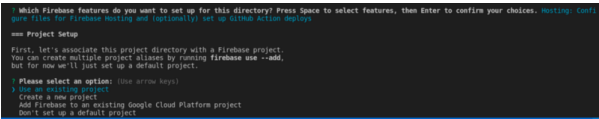


Ilustração: 27

As proximas perguntas importantes são public directory informe: **build/web**; single-page: No; deploy com GitHub: No.

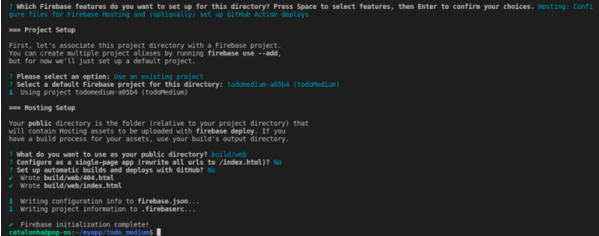


Ilustração: 28

Ficando nosso arquivo de firebase.json desta forma.

```
1 {
2   "hosting": {
3     "public": "build/web",
4     "ignore": [
5       "firebase.json",
6       "**/*.*",
7       "**/node_modules/**"
8     ]
9   }
10 }
```

Temos agora as configurações do App Flutter Android e do App Flutter Web. Isto pode ser visto:

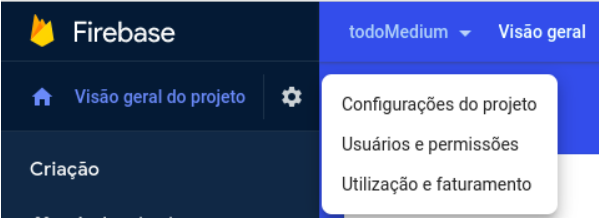


Ilustração: 29.1

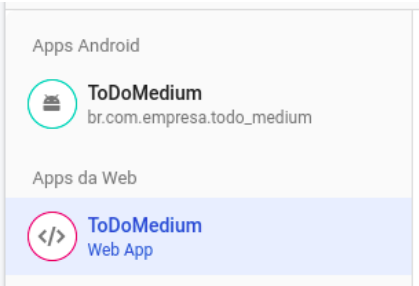


Ilustração: 29.2

Mas ainda falta uma etapa para que possamos compilar o projeto localmente e via remoto para fazer login via senha/email e google. Para isto vá nesta página: <https://console.cloud.google.com/apis/credentials> e escolha seu projeto.

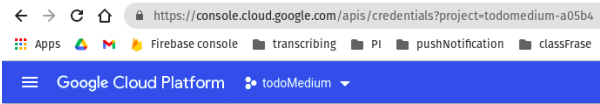


Ilustração: 30

No item OAuth 2.0 edite o Web Client. No lapis a direita, antes da lixeira. Um pouco antes no ícone de copiar/colar você também pode pegar o valor de ID do cliente Web para a linha 2. Lembre: `<meta name="google-signin-client_id" content="...">`

IDs do cliente OAuth 2.0

<input type="checkbox"/>	Nome	Data da criação ↓	Tipo	ID do cliente	Ações
<input type="checkbox"/>	Android client for br.com.empresa.todo_medium (auto created by Google Service)	10 de mar. de 2022	Android	[REDACTED]	[Icones]
<input type="checkbox"/>	Web client (auto created by Google Service)	10 de mar. de 2022	Aplicativo da Web	[REDACTED]	[Icones]

Ilustração: 31

Você precisa acrescentar a solicitação de <http://localhost:7357> e para <https://seuprojeto.web.app> e salvar esta alteração.

Origens JavaScript autorizadas ?

Para usar com solicitações de um navegador

URIs 1 *

URIs 2 *

URIs 3 *

URIs 4 *

URIs 5 *

+ ADICIONAR URI

Ilustração: 32

Executando para Flutter Android —emulator

Assim poderemos compilar o projeto para Flutter Android apenas teclando F5 no VSCode

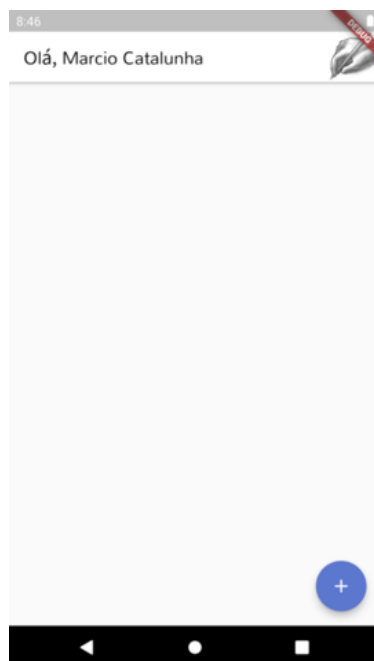


Ilustração: 33

Executando para Flutter Web—Localmente

E também poderemos compilar o mesmo código para Flutter Web com os seguintes comandos:

```
$ flutter build web—web-renderer html
```

```
catalunha@pop-os:~/myapp/todo_medium$ flutter build web --web-renderer html
👉 Building with sound null safety 👈
Compiling lib/main.dart for the Web... 58.2s
```

Ilustração: 33

```
$ flutter run -d chrome—web-hostname localhost—web-port 7357
```

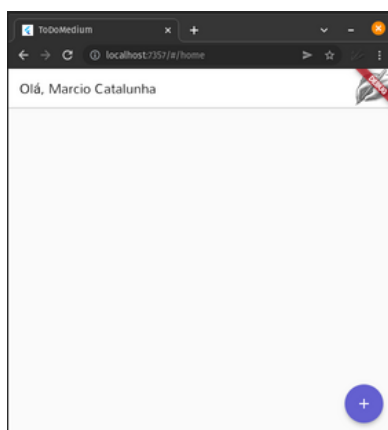


Ilustração: 34

Executando para Flutter Web—remotamente

Como tivemos sucesso na execução web localmente vamos apenas enviar para o firebase hosting com o comando e sua saída logo em seguida:

```
$ firebase deploy
```

```
catalunha@pop-os:~/myapp/todo_medium$ firebase deploy
=== Deploying to 'todomedium-a05b4'...

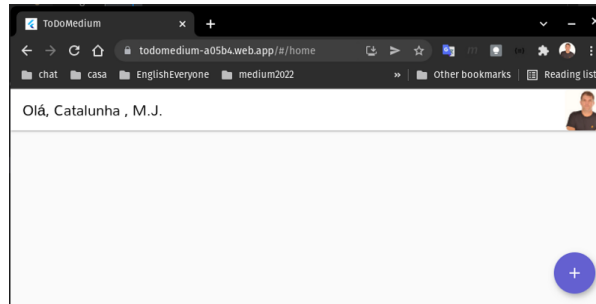
i deploying hosting
i hosting[todomedium-a05b4]: beginning deploy...
i hosting[todomedium-a05b4]: found 21 files in build/web
✓ hosting[todomedium-a05b4]: file upload complete
i hosting[todomedium-a05b4]: finalizing version...
✓ hosting[todomedium-a05b4]: version finalized
i hosting[todomedium-a05b4]: releasing new version...
✓ hosting[todomedium-a05b4]: release complete

✓ Deploy complete!

Project Console: https://console.firebase.google.com/project/todomedium-a05b4/overview
Hosting URL: https://todomedium-a05b4.web.app
```

Ilustração: 35

Após a execução você já pode acessar o app na web. Conforme visto a seguir.



Esta é a minha forma de desenvolver um App Flutter para Android e disponibilizar o mesmo app em Flutter Web no Firebase hosting.

Mas sinceramente gostaria de:

1. Saber da sua forma de fazer isto ?
2. Ou como a minha forma poderia ser melhorada ou otimizada no contexto do firebase hosting ?
3. Como realizar isto via GitHub Action ?
4. Ou até se tem outras formas de publicar esta app Flutter Web ?
5. Conhecer outros tutoriais nesta área !

Conclusão

Deu certo para este momento 2022-03-11. Mas foi fruto de alguns meses de tentativa e erro. E vamos debater o assunto. Pois melhor que implementar em implantar.

O mais importante...

Não somos apenas programadores. Somos pessoas, filhos/as, irmãos/ãs, maridos/esposas, pais/mães, tios/tias, colegas, amigos, etc. E em tudo isto precisamos revelar o amor ao próximo. E creio que este amor para ser completo e pleno vêm de Deus. E Deus é bom, e vale a pena buscar andar em seus caminhos. E Jesus Cristo é uma necessária referência para nossas vidas. Sem estes absolutos nossa vida, em alguns momentos, torna-se má e vazia. Então busquemos estar mais próximos de Deus a cada dia. Que a paz esteja com você e sua família.

Muito obrigado por chegar até aqui...

