



Firebase

Usabilidade, Desenvolv. Web, Mobile e Jogos

Prof. Gustavo Custodio
gustavo.custodio@ulife.com.br



firebase

Firebase

- Firebase é um conjunto de ferramentas para criar **aplicações na nuvem**.
 - Bancos de dados;
 - Autenticação;
 - Hospedagem;
 - etc.

Firebase

- O Firebase segue o padrão ***Backend as a Service (BaaS)***.
 - Não há necessidade de se preocupar com detalhes do *backend* quando cria sua aplicação.
 - Podemos focar apenas em aspetor do *frontend*.

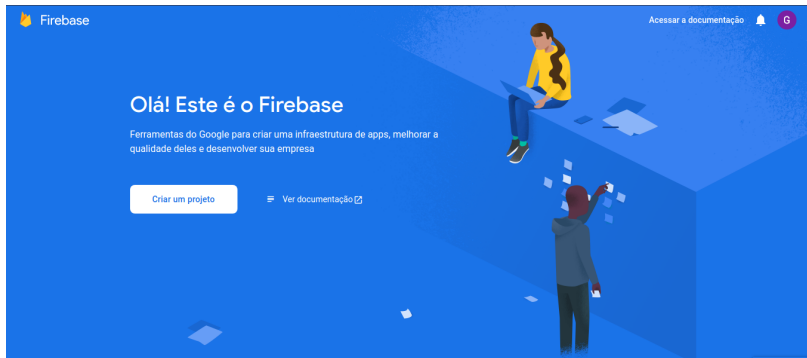


firebase

Configurando um App com Firebase

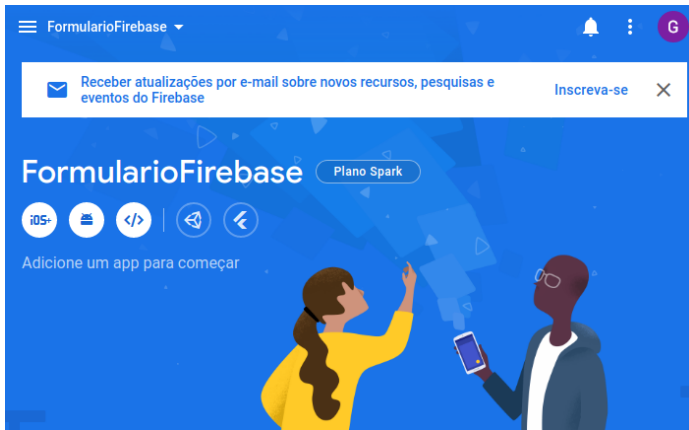
Configurando

- Entre em <https://firebase.google.com>



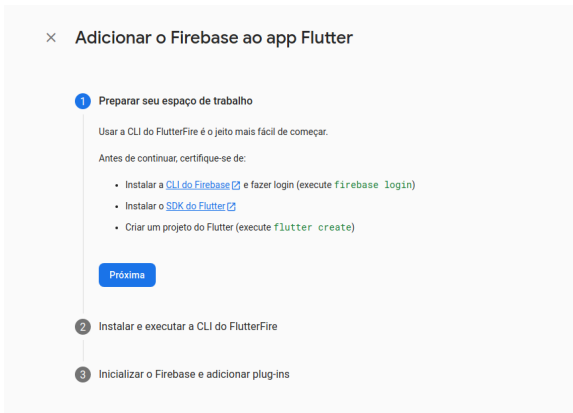
Configurando

- Crie um aplicativo chamado FormularioFirebase.



Configurando

- Clique no botão com o símbolo do flutter e você verá a seguinte tela:



- Clique em Próximo.

Configurando

- Instale a CLI (*Command Line Interface*) do Firebase.
 - Ela pode ser instalada pelo npm
 - ou baixando o arquivo binário autônomo (faremos isso).

binário autônomo

npm

Para fazer o download e executar o binário da Firebase CLI, siga estas etapas:

1. Faça download do [binário do Firebase CLI para Windows](#) .
2. Acesse o binário para abrir um shell onde você pode executar o comando `firebase` .
3. Continue a [fazer login e testar a CLI](#) .

Configurando

- Clique duas vezes no arquivo e você verá uma tela como essa:

```

=====
  Firebase CLI
=====

Welcome to...

#####  #####  #####  #####  #####  #####  #####  #####
##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##
#####  ##  #####  #####  #####  #####  #####  #####  #####
##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##
#####  #####  #####  #####  #####  #####  #####  #####  #####
##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##

=====

- Let's make sure your Firebase CLI is ready...
- Looks like your CLI needs to be set up.

- This may take a moment
- Setting up environment...

```

Configurando

- Neste console digite o comando:
 - `dart pub global activate flutterfire_cli`
 - O arquivo `flutterfire.bat` será gerado.
 - Copie o diretório em que o arquivo foi gerado.

Configurando

```
Downloading intl 0.17.0...
Downloading characters 1.2.1...
Downloading win32 2.7.0...
Downloading ffi 2.0.1...
Downloading clock 1.1.1...
Downloading source_span 1.9.1...
Downloading term_glyph 1.2.1...
Downloading json_annotation 4.7.0...
Downloading async 2.9.0...
Building package executables...
Built flutterfire_cli:flutterfire.
Installed executable flutterfire.
Warning: Pub installs executables into C:\Users\gusta\AppData\Local\Pub\Cache\bin, which is not on your path.
You can fix that by adding that directory to your system's "Path" environment variable.
A web search for "configure windows path" will show you how.
Activated flutterfire_cli 0.2.7.

> C:\Users\gusta\AppData\Local\Pub\Cache\bin\flutterfire.bat
```

Configurando

- Crie um projeto novo usando o Flutter.
- No diretório principal do seu projeto, na linha de comando, digite:
 - `<diretorio>\flutterfire.bat configure --project=<projeto>`
 - O diretório correto é mostrado no console do Firebase.
 - Um arquivo `firebase_options.dart` será gerado.

Configurando

- Será mostrado um conjunto de opções:
 - Android;
 - iOS;
 - Web.
- Escolha Web e responda Sim (*Yes*) nas perguntas mostradas.

Configurando

- Ainda na pasta do seu projeto, baixe os arquivos `firebase_core` e `firebase_auth`.
- Digite na linha de comando na pasta do seu projeto:
 - `flutter pub add firebase_core`
 - `flutter pub add firebase_auth`

Testando

- Altere o código do seu arquivo main.dart:

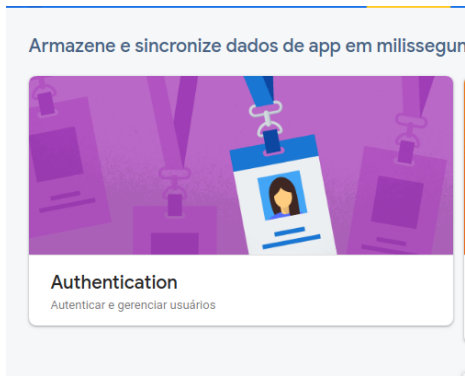
```
1  import 'firebase_options.dart';
2  import 'package:firebase_core/firebase_core.dart';
3
4  Future main() async {
5      await Firebase.initializeApp(
6          options: DefaultFirebaseOptions.currentPlatform,
7      );
8  }
```


Testando

- Tente rodar o app.
 - O **Firestore** não possui suporte para aplicações Windows, Linux, etc.
 - Rode como uma **aplicação Web**.
- Se nenhum erro ocorreu, a conexão com o *Firestore* foi um sucesso!
- Reverta o método `main` para sua forma anterior.

Configurando

- Dentro do seu console no *Firebase* é necessário autorizar o uso de serviços como a autenticação do *Firebase*.



Configurando

- Clique em *Authentication* e em “Vamos lá”.
- Selecione **E-mail/Senha**.
 - Clique em Ativar.

Configurando

Configurar provedor (Etapa 2 de 2)

 E-mail/senha ☒ Ativar

Permite que os usuários se inscrevam usando o endereço de e-mail e a senha deles. Nossos SDKs também fornecem verificação de endereço de e-mail, recuperação de senha e componentes essenciais para alteração do endereço de e-mail. [Saiba mais](#)

Link do e-mail (login sem senha) ☐ Ativar

Cancelar **Salvar**

- Agora você pode usar **serviços de autenticação por e-mail e senha** do *Firebase*.



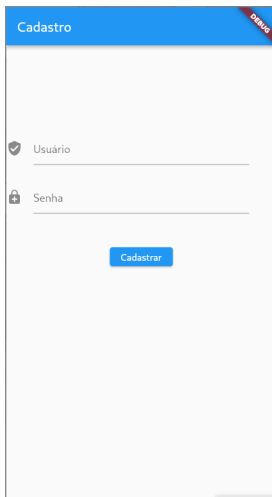
firebase

Criando um App com Firebase

Criando o App

- Vamos criar uma tela de cadastro de usuário.
 - Contendo usuário e senha.
 - Armazenaremos essas informações no Firebase.

Criando o App



A mobile application registration screen titled 'Cadastro' in a blue header bar. A red diagonal banner in the top right corner reads 'DRAFT'. The screen contains two input fields: 'Usuário' with a checkmark icon and 'Senha' with a lock icon. A blue 'Cadastrar' button is positioned below the fields. The bottom of the screen shows a portion of a mobile home indicator bar.

Cadastro

Usuário

Senha

Cadastrar

Criando o App

- Crie uma pasta chamada shared e dentro dela um arquivo `firebase_authentication.dart`.
- No topo do arquivo coloque:

```
1  import 'package:firebase_auth/firebase_auth.dart';
```


Criando o App

- No mesmo arquivo crie a classe:

```
1  class FirebaseAuthentication {  
2      // Instância do firebase  
3      final FirebaseAuth _firebaseAuth = FirebaseAuth.instance;  
4  }
```

Criando o App

- Crie uma função para cadastrar um usuário.

```
1 Future<String?> createUser(String email, String password) async {
2     try {
3         UserCredential credential = await _firebaseAuth
4             .createUserWithEmailAndPassword(email: email, password: password);
5         if (credential.user == null) {
6             return null;
7         }
8         return credential.user!.uid;
9     } on FirebaseAuthException {
10         return null;
11     }
12 }
```

Criando o App

- Crie um arquivo chamado `cadastro.dart` na pasta `lib` contendo os seguintes *imports*:

```
1  import "package:flutter/material.dart";
2  import "shared/firebase_authentication.dart";
3  import "package:firebase_core/firebase_core.dart";
4  import "firebase_options.dart";
```

Criando o App

- Crie um StatefulWidget chamado **FormularioCadastro**.

- Adicione dois Controllers na classe e um atributo mensagem.

```
1  class _FormularioCadastroState extends State<FormularioCadastro> {  
2  
3      final TextEditingController txtUserName = TextEditingController();  
4      final TextEditingController txtPassword = TextEditingController();  
5      String _mensagem = "";
```

Criando o App

- Ainda dentro dessa classe, crie quatro métodos diferentes para construir Widgets.
 - `_criarTxtUserName(context);`
 - `_criarTxtSenha(context);`
 - `_criarBotaoCadastro(context);`
 - `_criarCampoMensagem(context).`

Criando o App

```
1  Widget _criarTxtUserName(BuildContext context) {
2    return Padding(
3      padding: EdgeInsets.only(top: 128, right: 40),
4      child: TextFormField(
5        controller: txtUserName,
6        keyboardType: TextInputType.emailAddress,
7        decoration: InputDecoration(
8          hintText: "Usuário",
9          icon: Icon(Icons.verified_user),
10        ),
11      ),
12    );
13  }
```

Criando o App

```
1  Widget _criarTxtSenha(BuildContext context) {  
2    return Padding(  
3      padding: EdgeInsets.only(top: 24, right: 40),  
4      child: TextFormField(  
5        controller: txtPassword,  
6        keyboardType: TextInputType.emailAddress,  
7        obscureText: true,  
8        decoration: InputDecoration(  
9          hintText: 'Senha',  
10         icon: Icon(Icons.enhanced_encryption)),  
11      ),  
12    );  
13  }
```

Criando o App

```
1  Widget _criarBotaoCadastro(BuildContext context) {  
2    return Padding(  
3      padding: EdgeInsets.only(top: 50),  
4      child: ElevatedButton(  
5        onPressed: (){},  
6        child: Text('Cadastrar'),  
7      ),  
8    );  
9  }
```


Criando o App

```
1  Widget _criarCampoMensagem(BuildContext context) {  
2    return Text(  
3      _mensagem,  
4      style: TextStyle(  
5        fontSize: 16,  
6        color: Theme.of(context).primaryColorDark,  
7        fontWeight: FontWeight.bold),  
8    );  
9  }
```

Criando o App

- Adicione esses widgets em uma coluna no método build

```
1  Widget build(BuildContext context) {  
2    return Scaffold(  
3      appBar: AppBar(  
4        title: Text("Cadastro"),  
5      ),  
6      body: Column(  
7        children: [  
8          _criarTxtUserName(context),  
9          _criarTxtSenha(context),  
10         _criarBotaoCadastro(context),  
11         _criarCampoMensagem(context),  
12       ],  
13     ),  
14   );  
15 }
```

Criando o App

- No arquivo `main.dart` mude o método `build` para invocar o formulário:

```
1  import 'cadastro.dart';
2
3  @override
4  Widget build(BuildContext context) {
5      return MaterialApp(
6          title: 'Flutter Demo',
7          home: FormularioCadastro(),
8      );
9  }
```

Criando o App

- No início do `FormularioCadastro` adicione mais um atributo:
 - `FirebaseAuthentication? auth;`
- Quando o app é iniciado, a inicialização do *Firebase* deve ser realizada.
 - Adicionamos essa inicialização no método `initState`.

Criando o App

```
1      @override
2      void initState() {
3          Firebase.initializeApp(
4              options: DefaultFirebaseOptions.currentPlatform
5          ).whenComplete(() {
6              auth = FirebaseAuth();
7              setState(() {});
8          });
9      super.initState();
10 }
```

Criando o App

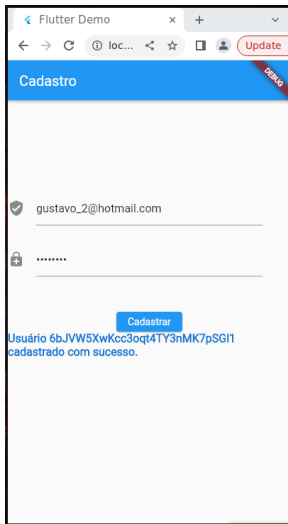
- Agora adicione o código dentro do onPressed do botão criado.

```
1
2  onPressed: (){
3    String userId = "";
4    auth!.createUser(txtUserName.text, txtPassword.text).then((value) {
5      if (value == null) {
6        setState(() {
7          _mensagem = 'Erro de cadastro.';
8        });
9      }else {
10        userId = value;
11        setState(() {
12          _mensagem = 'Usuário $userId cadastrado.';
13        });
14      }
15    });
16  }
```

Criando o App

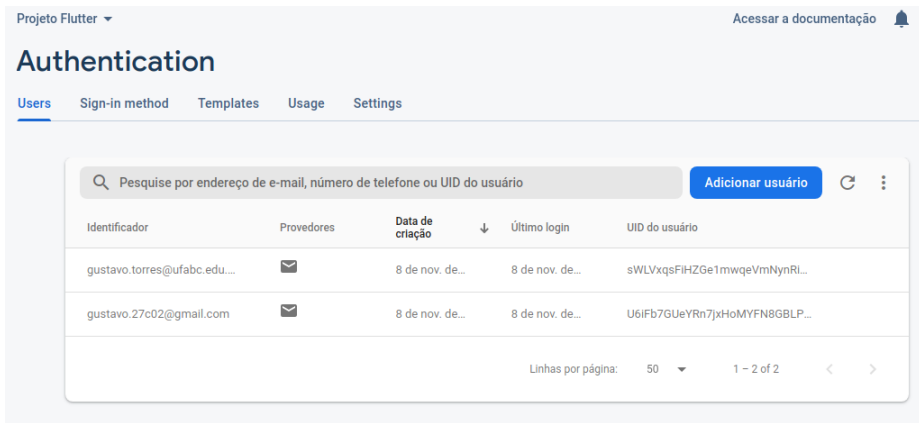
- Essa função tenta cadastrar um novo usuário utilizando o serviço de autenticação do *Firebase*.
 - Se o usuário é cadastrado, o campo de texto mostra que a operação foi um sucesso.
 - Se o cadastro falha, o usuário também recebe um *feedback*.

Criando o App



Criando o App

- É possível enxergar no console do Firebase todos os usuários cadastrados.
 - Basta clicar em *Authentication* e em *Users*.



The screenshot shows the Firebase Authentication console for a project named "Projeto Flutter". The "Users" tab is selected, displaying a table of registered users. The table has columns for Identifier, Providers, Date of creation, Last login, and User ID. Two users are listed: "gustavo.torres@ufabc.edu..." and "gustavo.27c02@gmail.com".

Projeto Flutter ▾ [Acessar a documentação](#) 🔔

Authentication

[Users](#) [Sign-in method](#) [Templates](#) [Usage](#) [Settings](#)

🔍 Pesquise por endereço de e-mail, número de telefone ou UID do usuário [Adicionar usuário](#) ↺ ⋮

Identificador	Provedores	Data de criação	↓	Último login	UID do usuário
gustavo.torres@ufabc.edu...	✉	8 de nov. de...		8 de nov. de...	sWLVxqsFIHZGe1mwqeVmNynRI...
gustavo.27c02@gmail.com	✉	8 de nov. de...		8 de nov. de...	U6iFb7GUeYRn7jxHoMYFN8GBLP...

Linhas por página: 50 ▾ 1 – 2 of 2 < >

Criando o App

- Adicione a função de *login* no arquivo `firebase_authentication.dart`.

```
1  Future<String?> login(String email, String password) async {
2    try {
3      UserCredential credential = await _firebaseAuth
4        .signInWithEmailAndPassword(email: email, password: password);
5      if (credential.user == null) {
6        return null;
7      }
8      return credential.user!.uid;
9    } on FirebaseAuthException {
10     return null;
11   }
12 }
```

Criando o App

- Adicione um botão de login ao formulário.

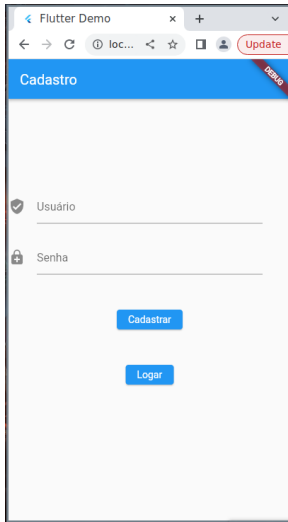
```
1 Widget _criarBotaoLogin(BuildContext context) {  
2   return Padding(  
3     padding: EdgeInsets.only(top: 50),  
4     child: ElevatedButton(  
5       onPressed: () {  
6         String userId = "";  
7         auth!.login(txtUserName.text, txtPassword.text).then((value) {  
8           if (value == null) {  
9             setState(() {  
10               _mensagem = 'Login inválido.';  
11             });  
12           } else {  
13             userId = value;  
14             setState(() {  
15               _mensagem = '$userId logado com sucesso.';  
16             });  
17           }  
18         });  
19       },  
20       child: Text('Logar'),  
21     ),  
22   );  
23 }
```

Criando o App

- No método build:

```
1      body: Column(  
2        children: [  
3          _criarTxtUserName(context),  
4          _criarTxtSenha(context),  
5          _criarBotaoCadastro(context),  
6          _criarBotaoLogin(context),  
7          _criarCampoMensagem(context),  
8        ],  
9      ),
```

Criando o App



Resumo

- Configuramos o *Firebase*.
- Utilizamos o *Firebase* para criar um usuário.
- Autenticamos o usuário criado com o *Firebase*.

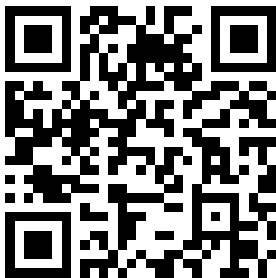
Referências

 Simone Alessandria and Brian Kayfitz.

Flutter Cookbook: Over 100 proven techniques and solutions for app development with Flutter 2.2 and Dart.

Packt Publishing Ltd, 2021.

Conteúdo



<https://gustavotcustodio.github.io/usabilidade.html>

Obrigado

gustavo.custodio@ulife.com.br