Desenvolvimento Multiplataforma com Flutter

**Uma imagem com texto, ecrã

Descrição gerada automaticamente**

|  |  |
| --- | --- |
| **Curso(s):** | Mestrado em Computação Móvel |
| **Unidade(s) Curricular(es):** | Seminário I |
| **Ano Letivo:** | 2021/2022 |
| **Docente:** | Carlos carreto |
| **Data:** | Vagner Bom Jesus – Nº 1701172 |

Índice

[Índice de figuras 3](#_Toc92183657)

[1 Introdução 4](#_Toc92183658)

[2 Visão geral do Flutter 5](#_Toc92183659)

[2.1 Resumo Histórico 5](#_Toc92183660)

[2.2 Arquitetura do Flutter 6](#_Toc92183661)

[2.3 Os Componentes do Flutter 7](#_Toc92183662)

[2.4 Anatomia de uma aplicação em Flutter 8](#_Toc92183663)

[2.5 Que recursos existem (documentação, tutoriais, exemplos)? 9](#_Toc92183664)

[3 Exemplos de Aplicações 10](#_Toc92183665)

[4 Exemplo prático 13](#_Toc92183666)

[4.1 Configurando o ambiente de desenvolvimento 13](#_Toc92183667)

[4.2 Instalação 13](#_Toc92183668)

[4.3 Instalação do Android Studio 14](#_Toc92183669)

[4.4 Instalação do Flutter 14](#_Toc92183670)

[4.5 Configurando plugin do Flutter 15](#_Toc92183671)

[4.6 Implementação 16](#_Toc92183672)

[4.7 Criação do projeto 17](#_Toc92183673)

[4.7.1 Core 18](#_Toc92183674)

[4.7.2 Pages 21](#_Toc92183675)

[4.8 Configurações 29](#_Toc92183676)

[4.8.1 Adicionar dependências da WebView 29](#_Toc92183677)

[4.8.2 Adicionando icon e nome a nossa aplicação 30](#_Toc92183678)

[*4.8.3* Configure o arquivo de configuração 31](#_Toc92183679)

[4.9 Aplicação final 31](#_Toc92183680)

[4.9.1 Aplicação SplashScreen 31](#_Toc92183681)

[4.9.2 Aplicação HomePage 33](#_Toc92183682)

[4.9.3 Aplicação na página do IPG 34](#_Toc92183683)

[5 Conclusões 35](#_Toc92183684)

[6 Referências 36](#_Toc92183685)

Índice de figuras

[Figura 1 - Arquitetura do Flutter (Flutter.dev, 2021) 6](#_Toc87913066)

[Figura 2 - Hierarquia de Widgets (Flutter.dev, 2021) 7](#_Toc87913067)

[Figura 3 - Estrutura de um projeto em Flutter 8](#_Toc87913068)

[Figura 4 - Empresas e organizações que trabalham com a Flutter 11](#_Toc87913069)

[Figura 5 - App Ipg para Android com java (Jesus, 2021) 16](#_Toc87913070)

[Figura 6 - Importar assets, Declarar Dependência 17](#_Toc87913071)

[Figura 7 - Estrutura base, theme 17](#_Toc87913072)

[Figura 8 – Configuração 30](#_Toc87913073)

[Figura 9 – SplashScreen 32](#_Toc87913074)

[Figura 10 – HomePage 33](#_Toc87913075)

[Figura 11 - IPG page 34](#_Toc87913076)

# Introdução

O presente documento descreve o projeto de investigação para apresentação do Workshop do curso de Mestrado em Computação Móvel do Instituto Politécnico da Guarda da unidade curricular de Seminário I ministrado pelo aluno Vagner Monteiro Vaz D’ Almeida Bom Jesus de nº 1701172, no ano letivo 2021/2022 com o tema Desenvolvimento Multiplataforma com o Flutter.

Com o desenvolvimento esporádico e popularidade dos smartphones, o mundo mobile tornou-se uma grande oportunidade de sucesso para programadores e empresas desenvolverem aplicações que estão disponíveis para quase o mundo inteiro.

Atualmente a existência de tecnologia na ária do desenvolvimento de aplicações de software multiplataforma permitem criar uma aplicação com uma única base de código em linguagem, que irá ser transformada para aplicação executáveis em múltiplos sistemas operativos como o Android e o iOS, proporcionando diversas vantagens tanto a nível de desenvolvimento como a manutenção da aplicação.

# Visão geral do Flutter

O Flutter é o kit de ferramentas da IU do Google para criar aplicações nativas e compilados para dispositivos móveis, web, desktop e incorporando uma única base de código, que tem o Dart como a principal linguagem de desenvolvimento, sendo o ponto de partida para o estudo e desenvolvimento em Flutter (Flutter, 2021).

O Dart é uma linguagem gratuita e de código aberto otimizada para o cliente e aplicativos rápidos em qualquer plataforma, sendo o seu objetivo é oferecer a linguagem de programação mais produtiva para o desenvolvimento multiplataforma (Dart, 2021). O Dart fornece uma linguagem e tempo de execução para suportar aplicativos Flutter, mas o Dart também oferece suporte a muitas tarefas principais do programador, como formatação, análise e teste de código.

## Resumo Histórico

A primeira versão do Flutter era conhecida com o codinome “sky”, com o intuito de encontrar uma melhor maneira de construir interfaces para o sistema operativo Android.

Foi apresentada em 2015 na Dart Developer Summit, onde foi possível ver o seu código-fonte Dart a executar num dispositivo Android. Em 4 de dezembro de 2018, o Flutter 1.0 foi lançado no evento Flutter Live Coding, sendo a primeira versão estável onde a aplicação desenvolvida foi integrada a plataforma Firebase da Google e utilizava a funcionalidade de aceder a câmara fotográfica do dispositivo.

Em 8 de setembro de 2021 foi lançado o Flutter na versão 2.5 onde foi adicionada suportes melhorando o desenvolvimento (Flutter, 2021) (Mediun, 2021).

## Arquitetura do Flutter

Flutter é projetado como um sistema extensível em camadas. Ele existe como uma série de bibliotecas independentes e cada biblioteca depende da camada inferior. Nenhuma camada tem acesso privilegiado às camadas abaixo e cada parte da camada estrutural é projetada para ser opcional e substituível (Flutter.dev, 2021).

Os principais componentes do Flutter incluem:

* Linguagem de programação Dart;
* Flutter Engine;
* Design-Specific Widgets com implementações pré-definidas para android (Google utilizando material Design), para iOS (Apple utilizando o Cupertiono Desing) e a WEB (utilizando o HTML e o CSS).

A Figura 1 a seguir demonstra o framework da arquitetura do Flutter:

Architectural
diagram

Figura 1 - Arquitetura do Flutter (Flutter.dev, 2021)

## Os Componentes do Flutter

Flutter é projetado para facilitar a criação de novos widgets e personalizar os widgets existentes. Os widgets formam uma hierarquia baseada em composição, onde cada widget herda atributos do seu pai. A Figura 2 demonstra essa hierarquia:

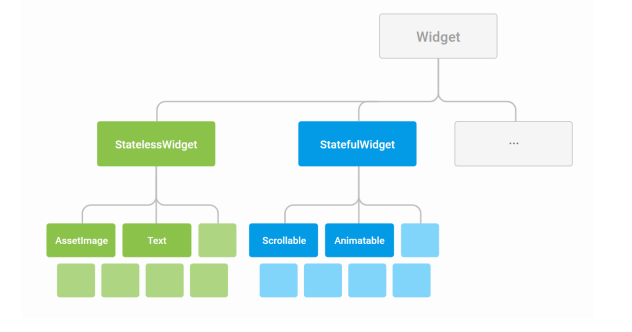


Figura 2 - Hierarquia de Widgets (Flutter.dev, 2021)

## Anatomia de uma aplicação em Flutter

Ao iniciar um projeto com Flutter é criada uma estrutura padrão, como mostra a **Erro! A origem da referência não foi encontrada.** abaixo:

* android: Contém ficheiros específicos da plataforma Android;
* iOS: Contém ficheiros específicos da Plataforma iOS;
* lib: Contém código-fonte Dart da aplicação, em que é possível criar pastas e subpastas, mas o ficheiro principal o “main.dart” deve estar na raiz do projeto;
  + main.dart: o ponto de partida de qualquer aplicação em Flutter, contém a chamada ao método “main()”;
* test: Pasta que coném os ficheiros de testes da aplicação;
* pubspec.yaml: Este ficheiro contém várias informações da aplicação, como o nome, versão descrisão, pacote/dependências, além do caminho para os ficheiros estáticos, como imagens, áudios, vídeos, gategorias de letras, entre outras;
* pubspec.lock: Contém metadados dos pacotes/dependências da aplicação;
* .idea: Contém os ficheiros com a extensão “.iml” utilizados apenas pela IDE IntelliJ;

Uma imagem com texto, eletrónica, preto

Descrição gerada automaticamente

Figura 3 - Estrutura de um projeto em Flutter

## Que recursos existem (documentação, tutoriais, exemplos)?

Documentações oficial para estudar flutter:

* **Dart Dev** – site oficial da linguagem Dart (<https://dart.dev/>);
* **Flutter Dev** – site oficial (<https://flutter.dev/>);
* **Pub Dev** – onde escontra os packages (<https://pub.dev/>);
* **Flutter Apprentice** – livro para apreender o desenvolvimento do Flutter (<https://www.raywenderlich.com/books/flutter-apprentice/>);
* **Tutoriais** – canal oficial no YouTube (<https://www.youtube.com/flutterdev>);
* **Github** – repositório público (<https://github.com/flutter>);

# Exemplos de Aplicações

Vale a pena utilizar o Flutter sendo baseado em três pilares importantes, desenvolvimento rápido, performance nativa, UI expressiva e flexível.

Por estas três características, o Flutter já pode ser considerado um forte candidato, mas a seguir estão mais algumas vantagens sobre a utilizar o Flutter para desenvolvimento de software:

* Criada e mantida pelo Google;
* Mais de 170 widgets (componentes) prontos para serem utilizados;
* Principais plugins para acesso a recursos do dispositivo (bateria, câmara, conectividades, webview, entre outros.) também mantidos pelo Google;
* O código-fonte Dart é executado diretamente no dispositivo;
* Possibilidade de desenhar todos os pixels no ecrã, tornando aplicação altamente customizados;
* Alta performance;
* Alta produtividade, pois a alteração no código fonte é refletida diretamente na aplicação executada no dispositivo ou emulador em até 0,5 segundos, caso precise reiniciar por completo a aplicação, o tempo necessário é menos de 2 segundos;

Algumas das maiores empresas do mundo estão a utilizar o Flutter para diversas aplicações (Flutter, 2021):

* BMW – o Flutter permitiu que construísse o aplicativo MY BMW para iOS e Android a partir de uma única base de código;
* Nubank – o Flutter ajudou numa missão de acabar com a complexidade dos bancos tradicionais e colocar o controle financeiro de volta na mão do usuário;
* Square Reader SDK – utiliza o plug-in Flutter permitindo que os programadores criem aplicativos para comerciantes que aceitam pagamentos personalizados;
* Tencent – usa o Flutter em toda a empresa para vários aplicativos. Incluindo ALTeacjer, Now Live, K12, Mr.Translator, QiDian e DingDang;
* Google – o Flutter ajuda a potencializar um número crescente de aplicativos do Google Assistente;

A Figura 4 demonstra mais algumas empresas que utiliza o Flutter:



Figura 4 - Empresas e organizações que trabalham com a Flutter

Exemplo de algumas aplicações móveis desenvolvidas com o Flutter apresentado abaixo:

[](https://www.youtube.com/embed/1y6Rpol6MMA?feature=oembed)

Figura - https://www.youtube.com/embed/1y6Rpol6MMA?feature=oembed

# Exemplo prático

## Configurando o ambiente de desenvolvimento

Para instalar e executar o Flutter, seu ambiente de desenvolvimento deve atender a estes requisitos mínimos:

* **Sistemas operacionais**: Windows 7 SP1 ou posterior (64 bits), baseado em x86–64.
* **Espaço em disco**: 1,64 GB (não inclui espaço em disco para IDE / ferramentas).
* **Ferramentas**: Flutter depende dessas ferramentas estarem disponíveis em seu ambiente. [Windows PowerShell 5.0](https://docs.microsoft.com/en-us/powershell/scripting/install/installing-windows-powershell) ou mais recente (pré-instalado com o Windows 10). [Git para Windows](https://git-scm.com/download/win) 2.x, com a opção **Usar Git no prompt de comando do Windows**. Se o Git para Windows já estiver instalado, certifique-se de executar gitcomandos no prompt de comando ou no PowerShell.

**Se a nossa máquina tem os requisitos mínimos, precisamos instalar os seguintes itens:**

* Flutter SDK;
* Android Studio e/ou um editor de texto com plug-ins de suporte ao Flutter (vamos utilizar o Android Studio como exemplo);
* Um emulador ou dispositivo para testarmos a aplicação.
* Configurar o plugin do Flutter no Android Studio (ou outro editor de textos com suporte ao Flutter).

## Instalação

Precisamos de uma IDE (ou editor de textos) para escrevermos nossos códigos e, com isso, desenvolvermos nossas aplicações. O Android Studio possui um suporte incrível para construir aplicações Flutter, desde criar um projeto até a compilação.

Dito isso, é essa ferramenta que utilizaremos como exemplo. Porém, ela não é a única existente, o [Visual Studio Code](https://code.visualstudio.com/) também possui um excelente suporte ao Flutter, porém veremos sua configuração em um outro artigo :)

Para instalação do Android Studio, é necessário baixar o instalador da página oficial, independente da plataforma que esteja utilizando. Você pode [baixar a última versão do Android Studio](https://developer.android.com/studio/) em sua página oficial. Após a cessar este link, basta clicar no botão “Download Android Studio” e aceitar os termos de licenciamento para que o download seja iniciado.

## Instalação do Android Studio

O processo de instalação do Android Studio no Windows é bem simples e segue o padrão Next > Next > Finish. Ao final deste processo, ele será instalado normalmente e estará funcional.

## Instalação do Flutter

A instalação no Windows é bem simples, necessitando apenas do Git e o próprio SDK do Flutter. Os passos para realizar esta instalação podem ser vistos abaixo:

* O Flutter é dependente do Git para funcionar. Por isso, é necessário que o Git esteja instalado na máquina. Para isso, basta [baixar o cliente do Git](https://git-scm.com/download/win) para Windows em sua página oficial. A instalação é no estilo Next > Next > Finish. Após o download do instalador do Git para Windows for concluído, basta clicar no instalador para que o processo de instalação seja iniciado;
* Certifique-se de que você esteja executando o Windows 7 SP1 64 bits ou superior;
* Baixe o arquivo ZIP da última versão do Flutter. O arquivo mais recente pode ser encontrado na página oficial do Flutter, dentro da seção “[Get the Flutter SDK](https://flutter.dev/docs/get-started/install/windows" \l "get-the-flutter-sdk" \o "Get the Flutter SDK" \t "_blank)”;
* Extraia o arquivo ZIP com o Flutter SDK para uma pasta onde não seja necessária a elevação de privilégios. A documentação do Flutter recomenda que o SDK seja extraído para C:\src\Flutter;
* Dentro da pasta onde o Flutter SDK foi instalado, localize o arquivo flutter\_console.bat e dê um duplo clique sob ele. Dessa maneira, o console do Flutter será inicializado e nós já poderemos rodar comandos para o Flutter;

Além disso, se quisermos colocar o Flutter CLI diretamente em nosso PATH (evitando termos que sempre abrir o Flutter Console de maneira manual), os seguintes passos são necessários:

* Vá em Painel de Controle > Contas de usuário > Contas de usuário > Variáveis de ambiente;
* Na seção de variáveis de ambiente de usuário, procure a variável chamada “Path”;
* Adicione à variável “Path” o caminho para a pasta flutter/bin. Não se esqueça de separar a nova entrada das entradas pré-existentes com um ponto-e-vírgula;
* Reinicie o Windows;
* Abra uma instância do PowerShell e digite o comando $ flutter doctor;
* De seguida executar o comando $ flutter doctor --android-licenses de modo a concordar com as licenças do Andoid;
* Com isso, o SDK do Flutter estará instalado e funcionando normalmente;

## ****Configurando plugin do Flutter****

Agora que já possuímos todo nosso ambiente configurado e funcional, precisamos instalar o plugin do Flutter ao Android Studio, é com ele que a IDE permitirá criar, gerenciar e executar apps Flutter. Sendo assim, no Android Studio é feito através de um plugin. Para instalar e configurar este plugin, siga os seguintes passos:

* Abra o Android Studio;
* Abra o Gerenciador de Plugins em File > Settings > Plugins (no caso do **Windows**);
* Clique em “Browse Repositories…”, selecione o plugin do Flutter e clique em “Install”;
* O Android Studio irá pedir a instalação do plugin do Dart também. Confirme a instalação deste plugin adicional;
* Aceite a reinicialização do Android Studio após a instalação do plugin.

Com isso, o nosso ambiente de desenvolvimento Flutter já está configurado e estamos aptos a desenvolver o nosso primeiro projeto.

## Implementação

O projeto consiste em criar uma aplicação para as plataformas Android e iOS, que permite integrar os principais sites do IPG – Instituto politécnico da Guarda e as suas plataformas de apoio a gestão escolar. Com esta aplicação em Flutter permitirá que os utilizadores de smartphones beneficiem de uma aplicação que está integrado os principais serviços do instituto.

Os sites que estarão presentes na aplicação são:

* **IPG** – Instituto Politécnico da Guarda
* **ESTG** – Escola Superior de tecnologia e gestão
* **ESTH** – Escola Superior de Turismo e hotelaria
* **ESECD** – Escola Superior de Educação, Comunicação e Desporto
* **ESS** – Escola Superior de Saúde
* **Moodle IPG** – Plataforma de Gestão Académica
* **Sigarra** **IPG** – Plataforma de Gestão Académica
* **Cantina** **IPG** – Sistema de aquisição de senha para cantinas dos Serviços de Ação Social

**NOTA**: Já a alguns anos um projeto idêntico foi feito por mim e funciona até os dias de hoje, mais erra uma aplicação somente para a plataforma android com java, como mostra na Figura 6 - App Ipg para Android com java (Jesus, 2021)Figura 6 abaixo.



Figura 6 - App Ipg para Android com java (Jesus, 2021)

## Criação do projeto

O projeto foi desenvolvido com a IDE Android Studio, Git para o controle de versões, após criar o projeto com o nome **workshops** no terminal do IDE executar o comando git init para inicializar o git e no projeto. E a estrutura do projeto inicial fica igual a **Figura 3.**

De seguida vamos importar os **assets**, e declarar o mesmo nas dependências do flutter como esta na Figura 7 abaixo.

Uma imagem com texto, captura de ecrã, interior

Descrição gerada automaticamente

Figura 7 - Importar assets, Declarar Dependência

De seguida criar a estrutura de pastas, definir a cor do theme e retirar o banner do debug, como apresentado na figura abaixo.

Uma imagem com texto

Descrição gerada automaticamente

Figura 8 - Estrutura base, theme

### Core

Criação de String estáticos ***app\_string.dart***:

class AppString {  
 static String get *splash* => "by: Vagner Bom Jesus";  
 static String get *app\_name* => "App IPG";  
 static String get *app\_title* => "IPG - Guarda";  
  
 static String get *app\_ipg* => "IPG";  
 static String get *app\_ipg\_deescricao* => "Istituto Politecnico da Guarda";  
 static String get *app\_ipg\_url* => "http://www.ipg.pt/website/";  
  
 static String get *app\_esth* => "ESTH";  
 static String get *app\_esth\_deescricao* => "Escola Superior de Turismo e Hotelaria";  
 static String get *app\_esth\_url* => "http://www.esth.ipg.pt/";  
  
  
 static String get *app\_ess* => "ESS";  
 static String get *app\_ess\_deescricao* => " Escola Superior de Saúde ";  
 static String get *app\_ess\_url* => "http://www.ess.ipg.pt/";  
  
 static String get *app\_estg* => "ESTG";  
 static String get *app\_estg\_deescricao* => "Escola Superior de Tecnologia e Gestão";  
 static String get *app\_estg\_url* => "http://www.estg.ipg.pt/";  
  
 static String get *app\_esecd* => "ESECD";  
 static String get *app\_esecd\_deescricao* => "Escola Superior de Educação, Comunicação e Desporto";  
 static String get *app\_esecd\_url* => "http://www.esecd.ipg.pt/";  
  
 static String get *app\_cantina* => "Cantina IPG";  
 static String get *app\_cantina\_deescricao* => "Sistema de Aquisição de senha para cantina dos Serviços de Ação Social";  
 static String get *app\_cantina\_url* => "https://ipg.luope.com/";  
  
 static String get *app\_moodle* => "Moodle IPG";  
 static String get *app\_moodle\_deescricao* => "Sistema de Gestão Acadêmica";  
 static String get *app\_moodle\_url* => "https://moodle.ipg.pt/";  
  
 static String get *app\_sigarra* => "Sigarra IPG";  
 static String get *app\_sigarra\_deescricao* => "Sistema de Gestão Acadêmica";  
 static String get *app\_sigarra\_url* => "https://cloud.sysnovare.pt/ipg/si\_main";  
  
}

Criação de Cores estáticas no ***app\_colors.dart***:

import 'package:flutter/material.dart';  
  
class AppColors {  
 static final Color *ipg* = Color(0xFFe40d2e);  
 static final Color *esecd* = Color(0xFFFF7F32);  
 static final Color *esth* = Color(0xFF84BD32);  
 static final Color *ess* = Color(0xFF00AFD7);  
 static final Color *estg* = Color(0xFF005f83);  
 static final Color *cantina* = Color(0xFFffc72c);  
 static final Color *moodle* = Color(0xFFf27f20);  
 static final Color *fundo* = Color(0xFFebf2ff);  
 static final Color *border* = Color(0xFFE1E1E6);  
 static final Color *purple* = Color(0xFF8257E5);  
 static final Color *white* = Color(0xFFFFFFFF);  
 static final Color *black* = Color(0xFF514766);  
 static final Color *grey* = Color(0xFF6E6680);  
 static final Color *lightGrey* = Color(0xFFA6A1B2);  
}

Criar String de images ***app\_images.dart***:

class AppImages {

static String get *logo* => "assets/images/iconipg.png";  
 static String get *ipg* => "assets/images/ipg0.png";  
 static String get *esth* => "assets/images/ipg2.png";  
 static String get *ess* => "assets/images/ipg3.png";  
 static String get *estg* => "assets/images/ipg4.png";  
 static String get *cantina* => "assets/images/ipg5.png";  
 static String get *esecd* => "assets/images/ipg6.png";  
 static String get *moodle* => "assets/images/moodle.png";  
 static String get *icon* => "assets/images/icons.png";  
}

Criar ***core.dart:***

export './app\_images.dart';  
export './app\_colors.dart';  
export './app\_string.dart';  
export './app\_card.dart';

Criar o Card ***app\_card.dart***:

import 'package:flutter/material.dart';  
import 'package:workshops/core/core.dart';  
class AppCard extends StatelessWidget {  
 final String title;  
 final String subTitle;  
 final VoidCallback onTap;  
 final Widget image;  
 const AppCard(  
 {Key? key,required this.title, required this.subTitle, required this.onTap, required this.image, }) : super(key: key);  
@override  
 Widget build(BuildContext context) {  
 return Container(margin: const EdgeInsets.all(16),: Material(  
 elevation: 2,  
 borderRadius: BorderRadius.circular(10), child: GestureDetector(  
 onTap: onTap,  
 child: Center(  
 child: Container(  
 padding: EdgeInsets.all(16),  
  
 decoration: BoxDecoration(  
 border: Border.fromBorderSide(BorderSide(color: AppColors.*border*)),  
 color: AppColors.*white*,  
 borderRadius: BorderRadius.circular(10)),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.center,  
 children: [  
 Expanded(  
 flex: 2,  
 child: Container(  
 height: 90,  
 width: 90,  
 child: image,  
 ),  
 ), SizedBox(  
 height: 8,), Expanded(  
 flex: 1,  
 child: Text(  
 title,  
 style: TextStyle(  
 color: Colors.*black*,  
 fontSize: 17,  
 fontWeight: FontWeight.*w600*,  
 )),), SizedBox(  
 height: 8,), Expanded(  
 flex: 1, child: Text(  
 subTitle,  
 textAlign: TextAlign.center,  
 style: TextStyle( color: Colors.*grey*,  
 fontSize: 11,  
 fontWeight: FontWeight.*normal*,),  
 ),  
 ),],), ),),),),);  
 }  
}

***main.dart:***

import 'package:flutter/material.dart';  
import 'package:workshops/pages/splash/splash\_page.dart';  
  
import 'core/core.dart';  
  
void main() {  
 runApp(const MyApp());  
}  
  
class MyApp extends StatelessWidget {  
 const MyApp({Key? key}) : super(key: key);  
  
 *// This widget is the root of your application.* @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 title: AppString.*app\_title*,  
 debugShowCheckedModeBanner: false,  
 theme: ThemeData(  
 primarySwatch: Colors.*deepPurple*,  
 ),  
 home: SplashPage(),  
 );  
 }  
}

### Pages

***splash/splash\_page.dart:***

import 'dart:async';  
  
import 'package:flutter/material.dart';  
import 'package:workshops/core/app\_colors.dart';  
import 'package:workshops/core/core.dart';  
import 'package:workshops/pages/home/home\_page.dart';  
class SplashPage extends StatefulWidget {  
 const SplashPage({Key? key}) : super(key: key);  
  
 @override  
 State<SplashPage> createState() => \_SplashPageState();  
}  
  
class \_SplashPageState extends State<SplashPage> {  
 @override  
 void initState() {  
 *// TODO: implement initState* super.initState();  
 startTime();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 body: initScreen(context),  
 );  
 }  
  
 startTime() async {  
 var duration = new Duration(seconds: 10);  
 return new Timer(duration, route);  
 }  
  
 route() {  
 Navigator.*pushReplacement*(context, MaterialPageRoute(  
 builder: (context) => HomePage()  
 )  
 );  
 }  
  
 initScreen(BuildContext context) {  
 return Scaffold(  
 body: Container(  
 color: AppColors.*white*,  
 child: Center(  
 child: Expanded(  
 child: Column(  
  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: <Widget>[  
 Container(  
 child: Image.asset(AppImages.*icon*,),  
 ),  
  
 SizedBox(  
 height: 20,  
 ),  
 CircularProgressIndicator(  
 backgroundColor: AppColors.*white*,  
 strokeWidth: 1,  
 )  
 ,  
 SizedBox(  
 height: 20,  
 ),  
 Text(  
 AppString.*splash*,  
 style: TextStyle(  
 fontSize: 20.0,  
 color: AppColors.*purple*,  
 fontWeight: FontWeight.*bold* ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 ),  
 );  
 }  
}

E de restantes da Pages são basicamente iguais ***sigarra/sigarra\_page.dart:***

import 'dart:io';  
import 'package:flutter/material.dart';  
import 'package:webview\_flutter/webview\_flutter.dart';  
import 'package:workshops/core/core.dart';  
class SigarraPage extends StatefulWidget {  
 const SigarraPage({Key? key}) : super(key: key);  
  
 @override  
 \_SigarraPageState createState() => \_SigarraPageState();  
}  
  
class \_SigarraPageState extends State<SigarraPage> {  
 late WebViewController controller;  
 double progress = 0;  
  
 @override  
 void initState() {  
 super.initState();  
 if (Platform.*isAndroid*) WebView.*platform* = SurfaceAndroidWebView();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return WillPopScope(  
 onWillPop: () async {  
 if (await controller.canGoBack()) {  
 controller.goBack();  
  
 return false;  
 } else {  
 return false;  
 }  
 },  
 child: Scaffold(  
 appBar: AppBar(  
 leading: IconButton(  
 icon: const Icon(Icons.*clear*),  
 onPressed: () {  
 controller.clearCache();  
 Navigator.*pop*(context);  
 }),  
 title: Text(AppString.*app\_moodle*),  
 actions: [  
 IconButton(  
 icon: const Icon(Icons.*chevron\_left*),  
 onPressed: () async {  
 if (await controller.canGoBack()) {  
 controller.goBack();  
 }  
 },  
 ),  
 IconButton(  
 icon: const Icon(Icons.*refresh*),  
 onPressed: () => controller.reload(),  
 ),  
 ],  
 ),  
 body: Column(  
 children: [  
 LinearProgressIndicator(  
 value: progress,  
 color: AppColors.*ipg*,  
 ),  
 Expanded(  
 child: WebView(  
 initialUrl: AppString.*app\_sigarra\_url*,  
 javascriptMode: JavascriptMode.unrestricted,  
 onWebViewCreated: (controller) {  
 this.controller = controller;  
 },  
 onProgress: (progress) =>  
 {setState(() => this.progress = progress / 100)},  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

Implementação do nosso home com diferentes módulos com o seguinte código na **home/home\_pages.dart**:

import 'package:flutter/material.dart';  
import 'package:workshops/core/core.dart';  
import 'package:workshops/pages/cantina/cantina\_page.dart';  
import 'package:workshops/pages/eseced/esecd\_page.dart';  
import 'package:workshops/pages/ess/ess\_page.dart';  
import 'package:workshops/pages/estg/estg\_page.dart';  
import 'package:workshops/pages/esth/esth.dart';  
import 'package:workshops/pages/ipg/ipg\_page.dart';  
import 'package:workshops/pages/moodle/moodle\_page.dart';  
import 'package:workshops/pages/sigarra/sigarra\_ipg.dart';  
  
class HomePage extends StatelessWidget {  
 const HomePage({Key? key}) : super(key: key);  
  
 @override  
 Widget build(BuildContext context) {  
 const mobileBreakpoint = 600;  
 return LayoutBuilder(  
 builder: (context, constraints) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text(AppString.*app\_title*),  
 ),  
 body: Container(  
 color: AppColors.*fundo*,  
 child: constraints.maxWidth < mobileBreakpoint  
 ? GridView.count(  
 crossAxisCount: 2,  
 crossAxisSpacing: 8,  
 mainAxisSpacing: 8,  
 children: [  
 *//IPG* AppCard(  
 image: Image.asset(AppImages.*ipg*),  
 title: AppString.*app\_ipg*,  
 subTitle: AppString.*app\_ipg\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const IpgPage()),  
 );  
 },  
 ),  
  
 *//ESTG* AppCard(  
 image: Image.asset(AppImages.*estg*),  
 title: AppString.*app\_estg*,  
 subTitle: AppString.*app\_estg\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const EstgPage()),  
 );  
 },  
 ),  
  
 *//ESTH* AppCard(  
 image: Image.asset(AppImages.*esth*),  
 title: AppString.*app\_esth*,  
 subTitle: AppString.*app\_esth\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const EsthPage()),  
 );  
 },  
 ),  
  
 *//ESECD* AppCard(  
 image: Image.asset(AppImages.*esecd*),  
 title: AppString.*app\_esecd*,  
 subTitle: AppString.*app\_esecd\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const EsecdPage()),  
 );  
 },  
 ),  
  
 *//ESS* AppCard(  
 image: Image.asset(AppImages.*ess*),  
 title: AppString.*app\_ess*,  
 subTitle: AppString.*app\_ess\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const EssPage()),  
 );  
 },  
 ),  
  
 *// Moodle* AppCard(  
 image: ClipRRect(  
 borderRadius: BorderRadius.circular(50),  
 child: Image.asset(AppImages.*moodle*)),  
 title: AppString.*app\_moodle*,  
 subTitle: AppString.*app\_moodle\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const MoodlePage()),  
 );  
 },  
 ),  
*// Sigara* AppCard(  
 image: Image.asset(AppImages.*ipg*),  
 title: AppString.*app\_sigarra*,  
 subTitle: AppString.*app\_sigarra\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const SigarraPage()),  
 );  
 },  
 ),  
  
 *// cantina* AppCard(  
 image: Image.asset(AppImages.*cantina*),  
 title: AppString.*app\_cantina*,  
 subTitle: AppString.*app\_cantina\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const CantinaPage()),  
 );  
 },  
 ),  
 ],  
 )  
 : GridView.count(  
 crossAxisCount: 3,  
 crossAxisSpacing: 8,  
 mainAxisSpacing: 8,  
 children: [  
 *//IPG* AppCard(  
 image: Image.asset(AppImages.*ipg*),  
 title: AppString.*app\_ipg*,  
 subTitle: AppString.*app\_ipg\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const IpgPage()),  
 );  
 },  
 ),  
  
 *//ESTG* AppCard(  
 image: Image.asset(AppImages.*estg*),  
 title: AppString.*app\_estg*,  
 subTitle: AppString.*app\_estg\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const EstgPage()),  
 );  
 },  
 ),  
  
 *//ESTH* AppCard(  
 image: Image.asset(AppImages.*esth*),  
 title: AppString.*app\_esth*,  
 subTitle: AppString.*app\_esth\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const EsthPage()),  
 );  
 },  
 ),  
  
 *//ESECD* AppCard(  
 image: Image.asset(AppImages.*esecd*),  
 title: AppString.*app\_esecd*,  
 subTitle: AppString.*app\_esecd\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const EsecdPage()),  
 );  
 },  
 ),  
  
 *//ESS* AppCard(  
 image: Image.asset(AppImages.*ess*),  
 title: AppString.*app\_ess*,  
 subTitle: AppString.*app\_ess\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const EssPage()),  
 );  
 },  
 ),  
  
 *// Moodle* AppCard(  
 image: ClipRRect(  
 borderRadius: BorderRadius.circular(50),  
 child: Image.asset(AppImages.*moodle*)),  
 title: AppString.*app\_moodle*,  
 subTitle: AppString.*app\_moodle\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const MoodlePage()),  
 );  
 },  
 ),  
  
 *// Sigara* AppCard(  
 image: Image.asset(AppImages.*ipg*),  
 title: AppString.*app\_sigarra*,  
 subTitle: AppString.*app\_sigarra\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const SigarraPage()),  
 );  
 },  
 ),  
  
 *// cantina* AppCard(  
 image: Image.asset(AppImages.*cantina*),  
 title: AppString.*app\_cantina*,  
 subTitle: AppString.*app\_cantina\_deescricao*,  
 onTap: () {  
 Navigator.*push*(  
 context,  
 MaterialPageRoute(  
 builder: (context) => const CantinaPage(),  
 ),  
 );  
 },  
 ),  
 ],  
 ),  
 ),  
 );  
 },  
 );  
 }  
}

## Configurações

### Adicionar dependências da WebView

Um plugin Flutter que fornece um widget WebView. No iOS, o widget WebView é apoiado por um WKWebView; No Android, o widget WebView é apoiado por um WebView (webview\_flutter 2.3.1, 2021).

|  |  |  |
| --- | --- | --- |
|  | Composição híbrida | Monitores virtuais |
| Suporte de teclado completo | sim | não |
| Suporte para Android SDK | Mais de 19 | 20+ |
| Desempenho total | Android 10+ | sempre |
| O padrão | não | sim |

**Usando monitores virtuais**

O modo está atualmente habilitado por padrão. Você deve, no entanto, certificar-se de definir o correto minSdkVersionem android/app/build.gradle(se anteriormente era inferior a 20):

android {

defaultConfig {

minSdkVersion 20

}

}

Nota: para os sites que não têm certificado ssh se configura como esta na Figura 9 abaixo:



Figura 9 – Configuração

### Adicionando icon e nome a nossa aplicação

**Ícones do Flutter Launcher** uma ferramenta de linha de comando que simplifica a tarefa de atualizar o ícone do iniciador do aplicativo Flutter. Totalmente flexível, permitindo que você escolha para qual plataforma deseja atualizar o ícone do iniciador e, se desejar, a opção de manter o ícone antigo do iniciador no caso de querer reverter em algum momento no futuro. (flutter\_launcher\_icons 0.9.2, 2021)

### Configure o arquivo de configuração

Adicione a configuração dos ícones do Flutter Launcher ao seu pubspec.yaml ou crie um novo arquivo de configuração chamado pubspec.yaml. Um exemplo é mostrado abaixo, tendo em conta os a indentação do file.

*# https://pub.dev/packages/flutter\_launcher\_icons* flutter\_launcher\_icons: ^0.9.2  
  
flutter\_icons:  
 android: true  
 ios: true  
 image\_path: "assets/images/iconipg.png"  
 adaptive\_icon\_background: "#ffffffff"  
 adaptive\_icon\_foreground: "assets/images/icons.png"

Depois de definir a configuração, tudo o que resta fazer é executar o pacote.

flutter pub **get**

flutter pub run flutter\_launcher\_icons:main

Para mudar o nome da App:

* iOS:
  + ios/Runner/info.plist

mudar o nome para “App IPG”

* Android:
  + android/app/src/main/AndroidManifest.xml

android: label (mudar o nome para “App IPG”)

**Nota**: para a geração de um launcher icon usei o gerador aqui ([link](https://romannurik.github.io/AndroidAssetStudio/icons-launcher.html#foreground.type=clipart&foreground.clipart=android&foreground.space.trim=1&foreground.space.pad=0.25&foreColor=rgba(96%2C%20125%2C%20139%2C%200)&backColor=rgb(68%2C%20138%2C%20255)&crop=0&backgroundShape=circle&effects=none&name=ic_launcher)).

***Para gerando Buid APK Flutter e no menu Build > Flutter > Build APK.***

## Aplicação final

### Aplicação SplashScreen



Figura 10 – SplashScreen

### Aplicação HomePage

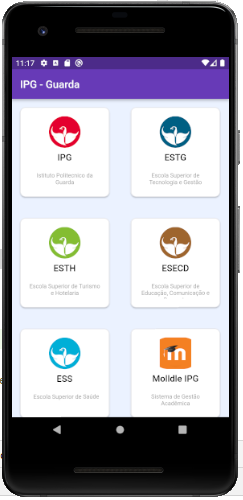


Figura 11 – HomePage

### Aplicação na página do IPG

Uma imagem com texto, captura de ecrã, monitor

Descrição gerada automaticamente

Figura 12 - IPG page

# Conclusões

Concluindo, o Flutter é uma ótima opção para desenvolver multiplataforma, com fácil reutilização e implementação dos widgets (componentes), e com esta mini implementação com único código para as plataformas Android e iOS respondeu muito bem a o que erra exigido.

O mesmo material está publicado no ***medium*** do ***@vagnerbomjesus*** dividido em 3 parte e o código completo disponível no ***github***: **https://github.com/VagnerBomJesus/workshops**

# Referências

*Dart*. (18 de 10 de 2021). Obtido de Dart: https://dart.dev

*Flutter*. (18 de Outubro de 2021). Obtido de Flutteer: https://flutter.dev/docs/whats-new

*Flutter*. (18 de 10 de 2021). Obtido de Flutter: https://flutter.dev/

Flutter.dev. (18 de Outubro de 2021). *Flutter*. Obtido de Flutter: https://flutter.dev/docs/resources/architectural-overview

*flutter\_launcher\_icons 0.9.2*. (15 de 11 de 2021). Obtido de pub.dev: https://pub.dev/packages/flutter\_launcher\_icons/install

Jesus, V. B. (15 de 11 de 2021). *Github*. Obtido de Github: https://github.com/VagnerBomJesus/IPG

*Mediun*. (18 de Outubro de 2021). Obtido de Flutter: https://medium.com/flutter/

*webview\_flutter 2.3.1*. (15 de 11 de 2021). Obtido de pub.dev: https://pub.dev/packages/webview\_flutter/install